ON QUANTITATIVE VALIDATION
AND OPTIMISATION OF
PHYSICALLY BASED DEFORMABLE
MODELS IN COMPUTER GRAPHICS

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Science and Engineering

2019

Matthew K. Banks

Department of Computer Science
Contents

Abstract 14
Declaration 15
Copyright 16
Acknowledgements 18

1 Introduction 21
   1.1 Discretisation of physical entities in computer graphics . . . . . . 22
   1.2 Discretisation of the equations of motion . . . . . . . . . . . . . 24
   1.3 Deforming models . . . . . . . . . . . . . . . . . . . . . . . . . . 25
   1.4 Structure of the thesis . . . . . . . . . . . . . . . . . . . . . . . 33
   1.5 Contributions . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
   1.6 Publications . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37

2 Background 38
   2.1 Chapter overview . . . . . . . . . . . . . . . . . . . . . . . . . . . 41
   2.2 Notation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 43
   2.3 Modelling deformations in computer graphics . . . . . . . . . . . 44
      2.3.1 Performance . . . . . . . . . . . . . . . . . . . . . . . . . . 45
      2.3.2 Stability . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 46
3 Quantitative Validation of Deformation Histories

3.1 Related work

3.1.1 Quantitative validation

3.1.2 Validation in surgical simulations

3.1.3 Validation in vehicle safety

3.2 QVDH software framework

3.2.1 Displacement measure component
A Stiffness Matrices 223

A.1 Stiffness matrix assembly for LFEM ..................................... 225
A.2 Stiffness matrix assembly for MSS ........................................ 229
   A.2.1 Direct stiffness method ............................................... 230

B Numerical Integration 232

B.1 Finite differences ............................................................. 232
B.2 Numerical integration schemes ............................................ 234

Word Count:
38835
List of Tables

1.1 Some example accuracy validation methods used in the computer graphics literature. .......................... 31

2.1 A summary of finite difference approximations to continuous derivatives. ................................................. 82

2.2 A summary of popular numerical integration schemes using in computer graphics. ................................. 85

5.1 Summary of the results of optimising two mass-spring stiffness parameters. ............................................ 162

5.2 Summary of the optimal model properties for PBDMs simulating 3D cantilever beams. ............................. 172

6.1 Restatement of the summary of optimised PBDM model properties for modelling the 3D cantilever. ............ 188

6.2 Two model property sets for PBD that favour performance ($P_{\text{perf}}$) and accuracy ($P_{\text{acc}}$). ............ 209
## List of Figures

1.1 How interactive computer graphics applications are related. 22
1.2 Example discretisations of a geometric model. 23
1.3 An example of visual artifacts due to poor PBDM modelling assumptions. 27
1.4 A visual example of numerical instability. 28
1.5 An example to demonstrate how visual plausibility is not an objective validation method. 30
1.6 A software framework is abstracted from the computer graphics literature to show how PBDMs were previously quantitatively validated. 32

2.1 Cantilever beams are modelled using different discretisations. 39
2.2 A PBDM’s update procedure. 42
2.3 The notation used to describe deformations. 43
2.4 The computational trade-off between performance and accuracy. 45
2.5 Refinement of the level of discretisation of a geometric model. 51
2.6 An example tetrahedralisation of a body for use in finite element calculations. 63
2.7 Visually comparing nonlinear, linear and corotational finite element methods. 75
2.8 A basic mass-spring system. ................................. 76
3.1 A trivial example to show the problem with using end-state validation. 93
3.2 The CORA metric. ........................................... 100
3.3 Measuring the agreement of two example signals. .............. 101
3.4 The component-based software framework used for quantitative validation of deformation histories. .................. 102
3.5 The setup of the 2D cantilever scenario. ....................... 108
3.6 Determining the minimum level of mesh refinement for accurately modelling a 2D cantilever beam. .................... 110
3.7 The visualisation of the 3D cantilever scenario. ............... 112
3.8 The convergence of the rating of the ISO18571 error metric with changes in the level of discretisation of the 3D cantilever. .. 114
3.9 The effect of varying the timestep for the reference simulation. 115
3.10 Spring configuration for a parallelepiped mesh. ............ 116
3.11 Constructing a mesh inside the original mesh to give a better mesh resolution per unit length. ......................... 117
3.12 Rating a mass-spring model integrated using an explicit integration scheme. ........................................ 119
3.13 Rating a mass-spring model integrated using an implicit integration scheme. ........................................ 122
4.1 The data flow of the component-based software framework. ... 131
4.2 A scenario component. ...................................... 133
4.3 A mesh component. ......................................... 136
4.4 How a mesh might be discretised. ............................ 136
4.5 A body component. ......................................... 137
4.6 A deformable model component. ............................ 138
4.7 A displacement measure component. .......................... 140
4.8 A discrete time signal component. ............................ 141
4.9 An error metric component. .............................. 141

5.1 The evolution of the QVDH software framework from Chapter 1
up until here. ........................................................... 145
5.2 Extending the software framework to perform model property optimisation. ................................. 156
5.3 Showing the effect that stiffness has on the deformation governed
by a mass-spring system. ............................................ 160
5.4 Optimising one mass-spring stiffness parameter. ................. 161
5.5 Optimising two mass-spring stiffness parameters. .................. 163
5.6 Optimising one mass-spring stiffness parameter when using an
implicit Euler scheme. ................................................. 164
5.7 Optimising one mass-spring stiffness parameter when using an
implicit trapezoidal scheme. ......................................... 165
5.8 The signals resulting from the optimised values of Figures 5.6 and
5.7 are plotted against the reference signal. ......................... 166
5.9 Optimising the solver iteration count of a position-based dynamics
model (integrated using BDF1). ......................................... 167
5.10 The signal resulting from the results of Figure 5.9. ................. 168
5.11 Optimising the solver iteration count of a position-based dynamics
model (integrated using BDF2). ......................................... 169
5.12 Optimising the distance constraint stiffness and solver iteration
count of a position-based dynamics model. ....................... 170
5.13 Optimising iteration count of an XPBD model. .................. 171
5.14 Summary plots of the signals from different optimised PBDMs. ... 172
5.15 The visualisation of the cloth scenario. .............................. 173
5.16 How springs are modelled across a cloth. .......................... 175
5.17 Optimising a position-based dynamics model for simulating cloth. 177
5.18 Showing how softer materials produce a better deformation over
time when the PBDM is damping the motion. ......................... 180

6.1 A 3D cantilever beam is tetrahedralised by TetGen [47]. ........... 194
6.2 A framework diagram for a 3D cantilever scenario using linear and
nonlinear finite element methods. ....................................... 195
6.3 Visualising the volumetric growth coming from the small displace-
ment assumption of the linear elasticity theory being violated. ... 197
6.4 Plotting the volumetric growth seen in Figure 6.3. .................. 198
6.5 Switching linear FEM for nonlinear FEM when volume is conserved. 199
6.6 Switching linear FEM for nonlinear FEM when volume is not
conserved. ....................................................................... 201
6.7 Measuring the global displacement gradient for a 3D cantilever beam. 203
6.8 Modelling a 3D cantilever beam and switching between LFEM and
NLFEM based on elastic potential energy levels. ..................... 206
6.9 Visualisation of the cloth scenario over time. ....................... 210
6.10 The framework diagram for a hanging cloth scenario with wind. .. 211
6.11 Plotting the signals of three PBD model property sets to show the
improvements made by switching. ...................................... 212
6.12 How switching improves the cloth simulation visually. .......... 216

A.1 A simple 1D mass-spring system from which we derive a stiffness
matrix. ................................................................. 231
Acronyms

**BDF** Backwards difference formula

**BDH** Bi-directional Hausdorff

**CG** Computer graphics

**CORA** Correlation and Analysis metric

**DH** Directed Hausdorff

**DTS** Discrete time signal

**EEARTH** Enhanced Error Assessment of Response Time Histories metric

**FEM** Finite element method

**IVE** Interactive virtual environment

**LFEM** Linear (solid mechanics equations numerically solved using the) finite element method

**MAE** Mean absolute error

**MSD** Mean square distance

**MSE** Mean square error

**MAX** Maximum error
MSS  Mass-spring system

NC  Normalised correlation

NLFEM  Nonlinear (solid mechanics equations numerically solved using the) finite element method

ODE  Ordinary differential equation

PBDM  Physically based deformable model

PBD  Position-based dynamics

PDE  Partial differential equation

PVD  Principle of virtual displacements

QVDH  Quantitative validation of deformation histories

RMSE  Root mean square error

SME  Subject matter expert

XPBD  Extended position-based dynamics
Abstract

On Quantitative Validation and Optimisation of Physically Based Deformable Models in Computer Graphics

Matthew K. Banks
A thesis submitted to the University of Manchester for the degree of Doctor of Philosophy, 2019

Physically based deformable models (PBDMs) in computer graphics (CG) simulate how virtual solid bodies respond under external loads in computer applications that must be computationally efficient enough so as to be user-interactive. Balance laws from the continuum mechanics theory govern the behaviour of solids. The modelling and simulation of this behaviour is traditionally performed using the finite element method (FEM), which solves to full accuracy and is therefore computationally expensive. This typically makes it unsuitable for IVEs and so different PBDMs are researched that are less expensive. These PBDMs make modelling assumptions that simplify the theory and their numerical results are less accurate than FEM.

In this thesis we present a novel software framework that allows us to quantify the accuracy of any deformation history of any PBDM as a result of its modelling simplifications. In the majority of previous studies, validation is qualitative (through visual plausibility), which is necessarily user subjective. We develop a novel, objective quantitative validation of deformation histories procedure (QVDH) to quantify the accuracy of any PBDM with an error between 0 and 1. We test QVDH in 3D cantilever and cloth scenarios, both of which are popular scenarios in the CG literature. The framework is shown to yield a high accuracy score for a simplified model that can be analytically derived from the reference model, indicating that the framework is reliable. We then extend the framework to optimise the model properties of PBDMs (that determine the material response) by minimising the error measured by QVDH. Results are in good agreement with analytically derived results, showing the effectiveness of the procedure. Finally, we use the framework to explore adaptive PBDMs - in particular PBDMs that switch to other PBDMs at runtime - and demonstrate how switching can successfully be implemented to increase the accuracy of a deformation according to QVDH.

The software framework is sufficiently general to be applicable to a wide variety of deformation scenarios. It has replaceable components that can help to improve the quality of the QVDH procedure. We believe that QVDH has many uses beyond those explored in this thesis.
Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.
Copyright

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made only in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication
and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=487), in any relevant Thesis restriction declarations deposited in the University Library, The University Library’s regulations (see http://www.manchester.ac.uk/library/aboutus/regulations) and in The University’s policy on presentation of Theses
Acknowledgements

This work was supported by the EPSRC. The results were obtained using Unreal Engine 4 game engine by Epic Games.

I would like to thank my supervisors Graham Riley and Andrew Hazel for their unparalleled guidance, necessary yet constructive criticisms and for the many joyous discussions over the past three years. You have made my PhD experience nothing short of excellent.

I would like to thank my fellow PhD cohort for their camaraderie throughout this degree. I would like to thank all my family and close friends for all of their support, particularly during the writing of this work and in preparation of the presentations I have had to give throughout. I would like to thank Chris for the many, many insightful discussions. Last, but my no means least, I would like to thank June for her love, support and never-ending patience.

I dedicate this work to my mother and father.
Acronyms

BDF  Backwards difference formula

BDH  Bi-directional Hausdorff

CG   Computer graphics

CORA Correlation and Analysis metric

DH   Directed Hausdorff

DTS  Discrete time signal

EEARTH Enhanced Error Assessment of Response Time Histories metric

FEM  Finite element method

IVE  Interactive virtual environment

LFEM Linear (solid mechanics equations numerically solved using the) finite element method

MAE  Mean absolute error

MSD  Mean square distance

MSE  Mean square error

MAX  Maximum error
MSS  Mass-spring system
NC  Normalised correlation
NLFEM  Nonlinear (solid mechanics equations numerically solved using the) finite element method
ODE  Ordinary differential equation
PBDM  Physically based deformable model
PBD  Position-based dynamics
PDE  Partial differential equation
PVD  Principle of virtual displacements
QVDH  Quantitative validation of deformation histories
RMSE  Root mean square error
SME  Subject matter expert
XPBD  Extended position-based dynamics
Chapter 1

Introduction

Computer graphics (CG) is a field that produces digitally synthesised image data to be visualised by a user. An interaction of a user with computer graphics data often occurs inside an interactive virtual environment (IVE), which may provide real-time simulation of particular physical phenomena [1][2][3][4]. Example IVEs are video games, surgical simulation scenarios and computer-aided engineering simulations such as in vehicle safety applications. These example fields are related within computer graphics through the relationship seen in Figure 1.1.

Simulations in IVEs must be computationally efficient enough so as to be updated in real-time: all necessary computations are complete in a small enough time so as to output data (in the form of rendering to a screen, audio, haptic force-feedback devices in the hands of the user e.g. surgical instruments, vehicle steering wheels, etc.) to the user in such a way that the user perceives the simulation as continuous. However, not all CG fields are real-time; animated movies require thousands of hours of rendering to produce a few of hours of screen animation. This thesis concerns simulations of deformations primarily found in real-time computer graphics applications in IVEs, although the methods presented are also applicable to non-real-time applications. In particular, we wish to quantify the accuracy of any simulated deformation and standardise a procedure to do so.
CHAPTER 1. INTRODUCTION

Figure 1.1: The relationship between CG applications inside IVEs is shown as a tree. CG is at the head of the tree and IVEs are subfields of CG applications.

In the real world, the motion of physical phenomena on macroscales is typically governed by continuous equations of motion that satisfy some physical conservation or balance laws [5][6]. To simulate motion we require a step to allow for the solution of these equations on a computer. This usually requires us to discretise both the entity (or solid body) being modelled and the equations of motion being solved. Discretisation is the process of representing continuous data as the union of a finite set of (often simpler) discrete data structures. In the next two Sections we introduce these two discretisation procedures.

1.1 Discretisation of physical entities in computer graphics

When we discretise physical entities in CG, there are three primary subfields of interest. These are geometric modelling, model rendering and model animation
1.1. DISCRETISATION OF PHYSICAL ENTITIES IN COMPUTER GRAPHICS

Geometric modelling comprises approximately representing the three- (or two-)
dimensional surface of an object using some discretisation of that surface into a
set of primitives. Most commonly this is through the use of discrete polygonal
meshes, where the boundary surfaces of a continuous entity are tessellated by
primitives such as triangles that meet at discrete vertices. Alternatively, point-
based modelling techniques such as particle systems can be used to model surfaces
that would otherwise be problematic to model using polygonal meshes (such as rain
or snow). As an example, a teapot is modelled at different levels of discretisation
using a polygonal mesh composed of triangles in Figure 1.2. To increase the level
of discretisation of these modelled surfaces is to increase the number of primitives
used along the surface.

Model rendering concerns the generation of an image from a model and how
light interacts with the model with a means to produce a desired artistic style.
Model animation concerns how the representation of the model changes over time. Animation can be performed “offline” in which the animation can be triggered to run its predetermined course. Alternatively, the animation can be performed at runtime. If the animation is physically based, the model deforms when it is subjected to external loads. This is the type of animation considered in this thesis.

1.2 Discretisation of the equations of motion

The motion of a physical body when subjected to external loads can be modelled using continuous differential equations. The solution procedure for solving these continuous equations of motion on a computer requires a discretisation of the following.

- The body being modelled, which is typically performed as seen in Section 1.1. The solutions to the governing equations are then obtained at the discretised vertices of the body.

- The continuous equations of motion, which provides a numerical approximation to the (continuous) differential equations and brings them into a finite-dimensional subspace. Solutions to these discrete equations are then typically found by a finite difference method [8] or a finite element method (FEM) [8].

The numerical approximations have both round-off and truncation errors, which are a consequence of using data structures which have only finite precision to store (real) values of infinite precision [9]. A finite difference approximation can be used to approximate the value of a continuous derivative. However, these approximations can introduce truncation errors of different orders of magnitude,
1.3. DEFORMING MODELS

depending on which terms are discarded from something such as a Taylor expansion, which can successively become magnified over further calculations and obscure the true solution. This magnification of the error is known as numerical instability [9], whereby small changes in initial data produce correspondingly large changes in the output data and the difference between successive approximations is unbounded. A more formal definition of numerical (in)stability is given in Section 2.3.2. We discuss different numerical approximations to derivatives in Section 2.6.1.

1.3 Deforming models

If the representation of the model changes with time so that its current configuration is different to its reference (initial) configuration (for example, if the orientation or size of some triangles in its polygonal mesh change), then the model is a deformable model. If the changes of nodal positions due to a deformable model are governed by some underlying physics, the model is a physically based deformable model (PBDM). The FEM used to solve the nonlinear equations that govern the motion of a solid at a sufficiently high resolution (abbreviated to NLFEM) is considered to be the state-of-the-art of accurate deformation modelling of solids [3] and is discussed in Section 2.5.1. These solutions typically require intensive computational procedures that produce a long time-to-solution that are not suitable for the real-time simulation of deformations. As such, a lot of research has been made in the effort to speed up the necessary computations of NLFEM [3], yet it still remains unsuitable for real-time simulations. Since interactivity in real-time simulations necessitates a short time-to-solution, deformation modelling was previously not possible in a real-time CG. Therefore only rigid, non-deforming models were used because for rigid bodies no consideration has to be made for how vertices might move relative to one another. However, the increase in power
of modern processors means that it is now possible to simulate PBDMs within CG in real-time [2]. It is most important in IVEs, however, that the interaction of a user with entities is robust and always responsive so as to maintain immersion (i.e. the quality of the experience of the user) [10]. In any interaction between the user and the virtual environment, the stability and performance of a simulation of a PBDM holds more importance than its accuracy [2].

Levels of computational performance, accuracy and stability are the distinguishing factors between PBDMs and the gains that can be made in computational performance are typically achieved through simplifications made to the equations of motion as a result of assumptions made on the type of deformation that will occur. For example, the governing equations of motion - which are generally nonlinear - can be linearised under the assumption that the deformation will be very small. This can greatly reduce the number of computations required to obtain the numerical solution at each update step. If this assumption is violated, however, the accuracy of the simulation of the deformation is subsequently reduced. This is visible in Figure 1.3. The most common computer-based representations of physics are in PBDMs including FEM, mass-spring systems (MSS - Section 2.5.5) and position-based dynamics (PBD - Section 2.5.6). As such, the motivation for researching and developing other PBDMs is therefore to produce PBDMs that perform well with a short time-to-solution and are accurate enough (i.e. have a sufficiently small amount of error from the true solution, as in the case with NLFEM) so as to appear realistic, all whilst maintaining their numerical stability.

The challenge of numerical stability

The advancement of a simulation in time requires numerical integration, which can introduce discretisation errors that lead to numerical instabilities, as mentioned previously. Figure 1.4 show the ramifications if numerical stability is not addressed.
1.3. DEFORMING MODELS

Figure 1.3: Three cantilever beams are simulated using FEM to solve the equations of motion. Of interest is the bottom beam, which is simulated under the assumption that any deformation will be sufficiently small. In this example, this assumption is violated; the volume is no longer conserved. These images are sourced from [11].

Numerical instability in simulating PBDMs is typically inherited from the numerical integration scheme chosen to advance the simulation forwards in time. A numerical integration scheme can be unconditionally stable, which means that the simulation will robustly react to every user input without giving a divergent solution as in Figure 1.4. This is preferable in IVEs as it improves the immersion of the user into that environment but comes with an increased computational performance cost. The topic of numerical integration methods is discussed in Section 2.6.

The challenge of computational performance

Real-time, interactive systems are bounded by time constraints that only provide tens of milliseconds for physical computations due to requirements to have a frame-update rate sufficient to produce a smooth visual presentation. Currently the frame computation time is between $\frac{1}{60}$ and $\frac{1}{30}$ seconds, which corresponds to between 30 and 60 frames-per-second (fps), which is the natural refresh rate of the
CHAPTER 1. INTRODUCTION

Figure 1.4: The problem of numerical stability during numerical integration is shown for cantilever beam simulations. The left, red beam is stable throughout, whilst the right, blue beam is not. The simulation of the right beam is stable at frame 5 (left screenshot) but becomes unstable by frame 10 (right screenshot). This is commonly referred to as “exploding” or “blowing up”, which are qualitative assessments of numerical stability. They describe the propagation and growth of errors in the underlying numerical method that give rise to instabilities in the numerical calculations.

colour televisions used to display the IVE [7]. If computations take too long then the application will not run at a real-time visualisation frequency (or frame-rate). This means that the image that is displayed to the screen will not be updated frequently enough so as to give the illusion of a real, time-evolving environment. The rationale behind a frame-rate of 60 fps is that it provides enough time to compute the necessary data for the next frame on modern day hardware but is also fast enough to look like a video to the human eye.

If deformations of higher complexity and sophistication are to be simulated then, in cooperation with improved hardware, the current methods of physical simulation must either be improved or optimised, or new methods discovered or adopted from other domains (which is a common approach in the CG literature). This thesis does not present new methodologies for physical simulation but instead presents a means to quantitatively evaluate the accuracy of such old and new
1.3. DEFORMING MODELS

models over a prescribed time period. In this thesis, we target architectures that are commonplace in video game consoles, which currently are the Xbox One and PlayStation 4, built with the 8-core Jaguar architecture of AMD [12].

The challenge of accuracy

The level of accuracy required for the simulated deformation is application domain-dependent. In surgical simulations, a high level of accuracy is crucial in order to adequately help train the user: if the simulation is not truly representative of reality, the training simulator is of little use. A PBDM, therefore, must seek to maximise accuracy and maximise computational performance (or, equivalently, minimise computation time) and stay numerically stable throughout. These factors have contrasting computational requirements. A more accurate model requires greater computational resources, reflecting the complexity of the nature of the deformation being modelled. If performance is important, then some degree of accuracy will typically have to be sacrificed to accommodate the performance improvements. Similarly, in engineering disciplines such as vehicle safety, accuracy is of extreme importance. In interactive virtual environments, numerical stability and computational performance have higher priority over accuracy.

While the PBDMs that are developed for use in real-time CG balance these three factors differently, currently the accuracy usually takes a lower priority. Further, the evaluations of the accuracy of the PBDM is found to be either objective or representative of the entire simulated deformation history but never both. This thesis presents a method that fulfills both of these criteria. In the real-time CG literature, accuracy evaluation is typically found to happen in one of two ways. The first of these is a qualitative evaluation, through the use of visual plausibility: does the entire simulation of the deformation look like it could happen in reality? Are there any strange visual artifacts that do not appear to be
correct? This evaluation is both a spatial (it looks correct with no visual artifacts) and a temporal (it looks correct throughout the simulation) evaluation of accuracy, but it is necessarily user-subjective: what might looks correct to one user might looks incorrect to another. A problem with a qualitative evaluation of accuracy for a cloth simulation is illustrated in Figure 1.5. In isolation, each deformation looks visually plausible, yet there is no knowledge of which is the more physically accurate of the two. The second evaluation is a quantitative evaluation using some error measure that quantifies the level of agreement between the deformation state produced by the PBDMs (the “test” data) and some reference data that is either pre-calculated, analytic or (real-life) experimental. Whilst this is standard engineering practice, currently the evaluation is only made at a single instance in time, which is usually the end of the simulation. Thus, this evaluation is both an objective and spatial evaluation but it is not a temporal evaluation since it does not evaluate the deformation history up until that single time instance. It could be the case that the test and reference data might quantitatively agree at this chosen final time instance, but quantitatively disagree up until that point.

Figure 1.5: Two PBDMs model the deformation of cloth [13]. Whilst both are visually plausible deformations, which deformation has greater physical accuracy?
1.3. DEFORMING MODELS

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Tested PBDM</th>
<th>Reference PBDM</th>
<th>Error Measure</th>
<th>Papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block of material</td>
<td>LFEM</td>
<td>real data</td>
<td>MAE, RMSE</td>
<td>[14]</td>
</tr>
<tr>
<td>Block of material</td>
<td>LFEM</td>
<td>real data</td>
<td>MSE, MAX</td>
<td>[15]</td>
</tr>
<tr>
<td>Beam</td>
<td>MSS</td>
<td>NLFEM</td>
<td>Visual</td>
<td>[16]</td>
</tr>
<tr>
<td>Cloth</td>
<td>PBD</td>
<td>NLFEM</td>
<td>Hausdorff</td>
<td>[17]</td>
</tr>
<tr>
<td>Breast tissue</td>
<td>MSS</td>
<td>real data</td>
<td>Median distance</td>
<td>[18]</td>
</tr>
<tr>
<td>Beam</td>
<td>NLFEM</td>
<td>NLFEM</td>
<td>Hausdorff</td>
<td>[19]</td>
</tr>
<tr>
<td>Beam</td>
<td>NLFEM</td>
<td>NLFEM</td>
<td>MAX</td>
<td>[19]</td>
</tr>
<tr>
<td>Brain tissue</td>
<td>FEM</td>
<td>real data</td>
<td>MAX</td>
<td>[20]</td>
</tr>
<tr>
<td>Rectal tissue</td>
<td>PBD</td>
<td>-</td>
<td>surgeon approval</td>
<td>[21]</td>
</tr>
<tr>
<td>Liver, gall bladder</td>
<td>MSS, PBD</td>
<td>real data</td>
<td>MAE</td>
<td>[22][23]</td>
</tr>
<tr>
<td>Liver, gall bladder</td>
<td>MSS, PBD</td>
<td>NLFEM</td>
<td>MAE, Hausdorff</td>
<td>[23]</td>
</tr>
<tr>
<td>Clinical MRI</td>
<td>LFEM</td>
<td>real data</td>
<td>MSD, NC</td>
<td>[24]</td>
</tr>
</tbody>
</table>

Table 1.1: This table shows the accuracy validation methods employed in examples of the real-time CG literature. A pattern emerges that consists of a selected scenario to model, some reference data, a tested PBDM that is being researched and developed and finally some kind of evaluation methodology. The evaluation methodology is either qualitative through visual plausibility or quantitative through an error measurement made at a single instance in time. MAE = mean absolute error, RMSE = root mean square error, MSE, mean square error, MAX = maximum error, MSD = mean square distance, NC = normalised correlation. All error measures employ the Euclidean distance metric. Of note is the subjective validation from surgeons found in [21]. This validation is subjective since it is SME-dependent. 15 SMEs (surgeons) were asked to interact with a surgical simulation of cutting and deforming rectal tissue before then being asked to complete a questionnaire that scored their opinion on the accuracy of the perceived deformations. The ratings concerning the accuracy of the deformations scored an average of 90%, showing SME approval and providing a SME-subjective measure, which, crucially, ought to be trustworthy.

Methods of accuracy evaluation, if any, in the literature are found to fit into these two types. A summary of the current state-of-the-art is presented in Table 1.1, which presents the validation methodology of various examples real-time CG literature pieces.

A pattern can be observed between these validation methodologies, which can further be abstracted to a component-based framework, as in Figure 1.6. We find
that two scenarios are often modelled in the CG literature: cantilever beams and cloths. We previewed the cantilever beam scenario in Figures 1.3 and 1.4 when we depicted numerical inaccuracies and we previewed the cloth scenario in Figure 1.5 when illustrating the problem with using visual plausibility as a validation method. They provide complex deformation behaviour and have simple mesh geometries that are easy to discretise. We model these two familiar scenarios in the tests conducted in Chapters 3, 5 and 6.

This component-based framework structure is one that is almost contained within SOFA [25]. SOFA is a software framework for the interactive simulation of physics-based models. There is particular emphasis made on medical applications and robotics. SOFA supports the simulation of any PBDM capable of producing a force field. Validation within SOFA almost follows the component-based framework of Figure 1.6. However, the error displacement measure is fixed to be the sum of absolute errors against the reference data. The reference data is loaded in from a file and has error measures at set time intervals. The test data of the PBDM is compared with the reference data at these time-marked values. Thus, states at multiple times can be compared. However, the error displacement component selection is not replaceable within the framework; it is hard-coded as the sum of absolute errors inside the MechanicalObject interface. In contrast, this thesis
presents a component-based software framework in which all components are easily replaceable without changing the functionality of the framework. The contents of the thesis follows.

1.4 Structure of the thesis

This thesis consists of seven Chapters. This introductory Chapter is followed by six further Chapters, of which four provide novel work. We summarise the contributions of this thesis in Section 1.5.

**Chapter 2** provides the background of the PBDMs used in this thesis that appear most commonly in the real-time CG literature. PBDMs must be numerically integrated forwards in time in order to numerically solve the discretised equations of motion. The equations of motion that are used are introduced and the most common numerical integration schemes are presented.

**Chapter 3** presents the *quantitative validation of deformation histories* (QVDH) procedure that is a means of obtaining a single, objective scalar quantity that serves as the error metric that validates a deformation history as being accurate or not. This procedure is the main motivation for this thesis. It comes from our literature review of real-time PBDMs and, in particular, the PBDMs not being validated objectively, quantitative and over a period of time (termed as the deformation history). Instead, validation is found to be either qualitative over a period of time or quantitative but only at a single instance in time. As well as presenting a novel means of quantitatively evaluating the deformation history, we present the appropriate extensions that have to be made to the component-based software framework as given in Figure 1.6. Our work in this chapter has been published (see Section 1.6).
Chapter 4 describes the component-based software framework introduced in Chapter 3. The framework is developed in order to obtain the results found in this thesis. All components that support the QVDH procedure of Chapter 3 can be easily replaced within the framework, depending on the specific problem being addressed. Previous works involving frameworks that support validation do not allow this generality in the form of replaceable components. The simplicity of the components and their interactions make the framework easy to reproduce and integrate into software packages when evaluating existing PBDMs or developing future PBDMs.

Chapter 5 describes an optimisation framework that is used to match the model properties of any PBDM against a reference PBDM using the QVDH procedure. The model properties of PBDMs often represent the material properties [6] of the body being modelled that govern how the material reacts under the influence of external loads. These properties are typically a challenge to adjust in order to obtain physically correct behaviour. We find that previous work in the real-time CG literature has focused on optimising the model properties of PBDMs against real-life reference data, with a view to having their PBDM accurately model complex material behaviour in a real-time setting. We present an optimisation method for optimising the model properties of any PBDM such that the resulting deformation of the PBDM is as accurate as it can be when compared against a reference data set using the QVDH procedure.

Chapter 6 introduces a mixed modelling technique that aims to increase the accuracy of the simulation of a deformation by using multiple PBDMs to deform the discretised body at runtime. Mixed modelling falls under the umbrella term of adaptive PBDMs. We find in the real-time CG literature that the utilisation
of mixed models is difficult to do well and identification of the points in space and time at which to switch models is a difficult problem to solve. Our main contribution here is to provide a software framework within which model switching can be implemented in order to support future research into the problem of when and where to switch. Our goal is to improve the accuracy of a PBDM (which we evaluate using the QVDH procedure) by switching to a more accurate PBDM in times of more complex deformations, which are the times at which the current PBDM might otherwise produce a greater error in accuracy. As examples, we use both the deformation gradient and the elastic potential energy of the system (both of these example measures quantify the amount of deformation and each term will be defined in Chapter 6) to prompt the switching to either a more or less accurate PBDM with a view to explore and allow the trade-off between computational performance and accuracy. We find that deformation gradient measures are successfully used in other adaptive PBDM methodologies outside of mixed models to refine the discretisation of a mesh. We show how it might successfully be used in the context of mixed models to improve the accuracy of a deformation (using the QVDH procedure implemented in our presented software framework).

Chapter 7 concludes the work of this thesis, discusses the limitations and shortcomings of the research and outlines some future work that is possible beyond this thesis.

1.5 Contributions

This thesis contains four contributions.

- The first of these contributions is the proposal of a metric in the form of a
single scalar value that serves as an error metric to quantitatively validate a
deformation history of a test PBDM as being accurate against some reference
PBDM. The reference PBDM is taken to be the ground truth: the state of
the body at each time instance of this deformation history is taken as being
physically correct, as is common engineering practice. The proposed error
metric is a value between \([0, 1]\) that quantifies the amount of error present in
the deformation history of the evaluated model against the chosen reference
model. Subsequently, the error metric may be used to directly compare the
accuracy of multiple different test PBDMs.

- The second contribution is a flexible component-based software framework
  that facilitates the construction of this single scalar value error metric.

- The third contribution is a means of optimising the material properties used
  in each tested PBDM to best-match that of a chosen reference PBDM. The
  optimisation makes use of the same error metric and thus the deformation
  history is taken into account during the independent parameter fitting.

- The fourth and final contribution is the identification of a switching mecha-
nism used for indicating when it is appropriate to switch between PBDMs
within the context of a single overall simulation. The goal of this is to
increase accuracy and/or reduce computation time. The proposed means of
switching is application-dependent, with the accuracy able to be evaluated
using the error metric. Improvements in accuracy are evaluated using the
quantitative validation procedure developed in this thesis.
1.6 Publications

The following publications have arisen from the research.

Published:

  This paper is [26] and forms the basis of Chapter 3.

In preparation:

- Physical Parameter Optimisation of Deformable Models Using Deformation Histories, MK Banks, AL Hazel, GD Riley.
  This paper forms the basis of Chapter 5.
Chapter 2

Background

In Figure 1.6 of Chapter 1 we abstracted and outlined a component-based framework from the CG literature that quantitatively validates a PBDM’s deformation based on its end state. This means that any error measure of a PBDM’s produced deformation is performed at a chosen single instance in time and that error measure quantifies the accuracy of the deformation. In Chapter 3 we build upon the component-based framework of Figure 1.6 and present a methodology to quantitatively assess the accuracy of a PBDM’s entire deformation history, rather than quantitatively assessing the accuracy at a single time instance of that deformation.

To successfully use the component-based framework outlined in Figure 1.6 and later in Chapter 3, we must instantiate all of the components and pass their data between one another. Two required components are PBDMs. The first of these PBDMs is the reference PBDM whose output deformation history is taken as the ground truth for the particular scenario being modelled and simulated. The second PBDM is a(ny) test PBDM whose output deformation history will be compared against the reference PBDM’s deformation history. We compare the two deformation histories to obtain a measure of accuracy. In this Chapter we
Figure 2.1: A cantilever beam is modelled using a co-rotated finite element method (CoRotFEM, introduced in Section 2.5.3) applied to two different mesh geometry configurations. As is visible in the right screenshot of the wireframe meshes, the mesh on the left is discretised into cubic unit cells, or elements (note that the backwards facing faces relative to the current camera have been culled and are omitted from the rendering process). The mesh on the right is discretised into tetrahedral elements. The CoRotFEM numerically solves the discretised equations of motion of Section 2.4 across each of these elements individually, therefore, nothing but the discretisation used is different. We can see that the simulations are not deforming in the same way. A change in both element type and the number of elements can lead to discretisation errors if the mesh is not refined enough into small enough elements in areas of more complex deformation. The area of more complex deformation here is along the lower surface that is bending.
introduce the most popular PBDMs in the CG literature.
2.1 Chapter overview

Before stating the contents of this Chapter, we first remark that the research and development of PBDMs in real-time CG necessarily must be:

- Computationally efficient in obtaining a solution so that we can interact with it inside a (real-time) virtual environment.

- Numerically stable (or robust); any interaction must not cause a divergent solution and break the simulation.

- Accurate enough so as to be realistic or plausible. Accuracy is typically of lesser importance than computational efficiency and robustness in IVEs but high accuracy is an attractive prospect nevertheless.

The motivation behind the research and development of new PBDMs in the CG literature is to obtain high levels of computational performance, high levels of accuracy and unconditional numerical stability of a PBDM’s deformation. We discuss each of these factors in Section 2.3. The rest of this Chapter introduces the most common PBDMs found in the CG literature that will be used later on in the results Sections of Chapters 3, 5 and 6. In Section 2.4 we state the equations that describe the motion of a continuum under applied external loads. In addition to discretising the continuum into nodal points, we must also discretise these equations of motion in order to numerically solve them at the discrete nodal points. We numerically solve the equations at each update step of the simulation to obtain the next (positional) state of discretised nodal points. In Section 2.5 we present the differences between the PBDMs commonly found in the CG literature that we will use in later Chapters to evaluate the accuracy of, to fit the model properties of and to switch between at runtime. All of the PBDMs presented provide solutions to the equations of motion, however we note that the differences between PBDMs
lies in the measure of deformation used and in the internal force computation, which describes how internal “particles” of the body being modelled interact with one another at a microscopic level. We illustrate the update procedure as a flowchart in Figure 2.2 that describes the update step that each PBDM performs.

Finally, we note that common to all PBDMs’ solution procedures is a numerical integration step. The most common numerical integration schemes are presented in Section 2.6.

Figure 2.2: The update procedure for each PBDM requires computation of the external forces at each node. Contact resolution would be handled as part of the external force computation. The internal force computation is what differentiates each PBDM. The numerical integration step is common to each PBDM and we can use a variety of numerical integration schemes that directly affect the performance, accuracy and stability of the simulation.

Turning our attention back to Table 1.1, we can see that it is common practice to use FEM (applied to solving the nonlinear equations that govern the motion of solid deformable bodies, i.e. NLFEM) as the reference PBDM. NLFEM is the topic of Section 2.5.1. We will also (therefore) use NLFEM as the instantiated reference PBDM component in the component-based software framework used to conduct the experiments in Chapters 3-6. In the Sections that follow this Section we introduce other popular PBDMs, namely linear FEM (LFEM, Section 2.5.2), corotational FEM (CoRotFEM, Section 2.5.3), mass-spring system (MSS, Section 2.5.5) and position-based dynamics approaches (PBD, Section 2.5.6). In particular, we present how the deformation is measured, how the internal forces are (thus) calculated and the procedure that we then follow to obtain the unknown
displacement field using the governing equations of motion.

2.2 Notation

We denote space using the (orthonormal) Cartesian basis vectors \( e_1, e_2, e_3 \). A position in space with coordinates \( (x_1, x_2, x_3)^T \) is denoted by the bold script, \( \mathbf{x} \), representing the vector from the global origin, \( \mathbf{0} \), to that position. The position vector of a material point on a body in its undeformed configuration is given by a lower case letter, for example, \( \mathbf{x} \). The corresponding material point in a deformed configuration is given by the upper case letter \( \mathbf{X} \). The displacement of such a material point is \( \mathbf{u} = \mathbf{X} - \mathbf{x} \). See Figure 2.3. The material point has velocity \( \mathbf{v} \) and acceleration \( \mathbf{a} \). Matrices, as rank-2 tensors, are expressed as capital letters unless otherwise specified, for example \( \mathbf{K} \in \mathbb{R}^{3 \times 3} \).

![Figure 2.3: A solid body is deformed. On the left is the unloaded, stress-free and undeformed configuration and on the right is a deformed configuration. A material point is marked with position vector \( \mathbf{x} \) that is displaced to \( \mathbf{X} = \mathbf{x} + \mathbf{u} \) after the deformation.](image-url)
2.3 Modelling deformations in computer graphics

As mentioned in the Chapter overview (Section 2.1), different PBDMs balance the three factors of computational performance, numerical stability and accuracy differently. We are interested in quantifying the accuracy of the deformation of a PBDM. In real-time CG applications, however, stability and performance are typically more important than accuracy. In the extreme case, some non-physically based models have been developed that knowingly sacrifice accuracy for the sake of performance gains, for example deformation based on shape-matching [27]. Whereas in surgical simulation, accuracy is understandably of the utmost importance [28]. To offer surgery training in an interactive virtual environment would save resources and move away from animal testing, provided that the simulations are faithful to what it is they are simulating [1]. We see that there is a real motivation for developing fast, accurate and stable PBDMs aside from making spectacular looking deformations in real-time for entertainment purposes. The three factors of high computational performance, accuracy and stability typically lead to opposing computational demands, whereby seeking to improve one tends to reduce at least one of the other two. The easiest trade-off comes from trading performance and accuracy. See Figure 2.4. Increasing the level of discretisation of the modelled continuum reduces any discretisation error (and therefore increases accuracy) but requires more computations at each solution step. Discretisation error is discussed further in Section 2.3.3. We discuss each of these three factors in more detail next and highlight the norms of the real-time CG literature.
2.3. MODELLING DEFORMATIONS IN COMPUTER GRAPHICS

2.3.1 Performance

The computational performance in a CG setting is measured as the frequency at which the geometry being modelled can be successfully rasterised for displaying to a screen, called the frame rate, and has units Hertz, or frames-per-second. The application runs in real-time if the algorithms and simulations that are used in the application can produce their results synchronously with the rasterisation process that outputs the image to screen, the audio that is produced or the haptic feedbacks that are relayed to the user through force-feedback devices. For humans, motion is perceived for at least 12Hz [29], whilst it is typical for video games to run at 30Hz at least, with the target frame rate currently 60Hz in video game settings. For physical interaction - through the use of haptic force feedback - the rates need to be at least 110Hz [30][31].

The challenge for real-time frame rates with PBDMs lies in the limited computational resources available to perform the necessary calculations. For any one frame in an interactive virtual environment, calculations must be performed for
each and every aspect of the application. This includes updates to the position and orientation of the camera, AI logic, non-physics-based animation, the user interface, physics and the graphics pipeline [7]. To simulate the motion of a deformable body in parallel to these other features of an IVE is a difficult problem to solve because the limited computational resources must be shared amongst all of them. In addition, modern household computer architectures (such as that found on Xbox One and PlayStation 4) are multi-core processors (currently four cores with simultaneous multithreading) equipped with vector registers [12]; parallelism is in the form of data parallelism (SIMD: single instructions acting across multiple data points in parallel) and through the use of a shared memory model. In a video game architecture, the graphics processing unit (GPU), which provides the ability for many instructions to be computed in parallel, is typically reserved for use in the graphics pipeline to help render complex scenes and so cannot be used for physical simulation. However, it is used in some real-time PBDM software [32]. Whilst architectures are increasing in computational power and can perform more numerical operations per update step, each feature of an application can always “do more” per update step to improve the experience in the IVE. For example, nicer graphics with more complex shaders, smarter AI and more complex physical interactions and deformations. So the best solution to produce more accurate, stable yet efficient deformations is not necessarily to increase the number of computational resources.

2.3.2 Stability

The deformation of a PBDM is stable if we can interact with the body in any way without the simulation of the deformation introducing numerical errors that obfuscate the true numerical solution (of the deformation) over time. In Section
1.2 we introduced the notion of instability as being when small changes to input data produces correspondingly large changes to output data, which in this context means that the solution is not sufficiently approximately accurate over time. More formally, we define numerical stability as follows. Suppose we have a function \( x(t) \) with given initial values \( x(t = 0) = x_0, x(t = 1) = x_1 \) are two solutions to a differential equation. Suppose also that \( y_i, z_i \) are the successive, time-evolving numerical approximates to \( x(t) \), starting from \( x_0 \) and \( x_1 \), respectively. Then the numerical method that approximates \( x(t) \) is numerically stable if, for each \( \epsilon > 0 \), there is a \( \delta > 0 \) sufficiently small such that

\[
|y_i - z_i| < \epsilon \quad \text{whenever} \quad |x_1 - x_0| < \delta.
\]

Otherwise, the method is numerically unstable.

Numerical instability might arise from the numerical integration scheme used to advance the simulation forwards in time. Numerical integration schemes are discussed in Section 2.6, where some are discussed as being computationally simple but only stable for sufficiently small time steps. We numerically integrate the equations of motion forwards in time, starting with Newton’s second law.

Newton’s Second Law of motion described the motion of a discrete particle with mass \( m \) and can be stated as:

\[
f = ma. \tag{2.1}
\]

This equation is a second order ordinary differential equation (ODE). The equation links the external loads, \( f \) to the accelerations, \( a \), of the discrete nodes of a deformable body between times \( t_1 \) and \( t_2 \) and can be solved numerically by first treating it as a system of two first order ODEs involving the nodal velocities, \( v \),
and displacements, $\mathbf{u}$:

$$
\mathbf{v}(t_2) = \mathbf{v}(t_1) + \int_{t_1}^{t_2} \mathbf{a}(\tau)d\tau = \mathbf{v}_1 + \int_{t_1}^{t_2} \frac{\mathbf{f}(\tau)}{m(\tau)}d\tau
$$

$$
\mathbf{u}(t_2) = \mathbf{u}(t_1) + \int_{t_1}^{t_2} \mathbf{v}(\tau)d\tau.
$$

When numerically integrating the ODEs, the properties associated with the chosen numerical integration scheme directly influence the performance, accuracy and stability of that PBDM.

In many cases the equations of motion govern stiff systems, the definition of which is not universally agreed upon [33]. A dynamical system is a stiff system if certain numerical methods must take small enough time step sizes in order to remain numerically stable. This numerical instability can arise when the solution rapidly varies depending on some of its input terms [34]. Mathematically, a constant coefficient linear system is stiff if all of its eigenvalues have a negative real part and the ratio between the largest a smallest eigenvalue is large, reflecting how a solution can rapidly vary depending on the values of its terms. Moreover, a system is stiff if the stability requirements, rather than the accuracy requirements, constrain the upper bound on the time step length.

In solving stiff systems, some numerical integration schemes may have a bounded region of stability inside which the numerical solution will not grow away from the true solution at each successive computation. Outside of this region, however, the integration method is unstable and the solution diverges away from the true solution. Visually, this phenomenon will show the body being modelled “exploding” or “blowing up” as the vertices of the mesh move in an undesired fashion, an example of which we saw in Figure 1.4. Computationally, the values being numerically integrated may eventually take the form not-a-number, or NaN.
to reflect this divergence. In an interactive virtual environment, if the simulation of a body becomes unstable, the rest of the environment can proceed and the simulation of the body can be reverted to a stable state. However, whilst doing so keeps the application running, the interrupted interaction that the user has with the simulated body will break the immersion of the user, violating the intended purpose of the simulation. Thus, the stability of a deformation simulation is of most importance within a virtual environment. *A-stability* of a numerical integration scheme is a desirable property when used to solve the equations of motion of a PBDM because the error in the solution does not grow [8], which is discussed in Section 2.6.

Alternatively, PBDMs might instead work directly on altering positions based on some predefined constraints that must be satisfied at every simulation step. That is, some PBDMs are not force-based, suffering from the possibility of conditional stability, but instead are *position-based*. A predictor step is performed that integrates forwards in time for the displacements (a computation that might be unstable), which is then followed by a correction step that employs positional constraints that prevent undesirable diverged displacements. This is the topic of Section 2.5.6. Whilst such PBDMs produce unconditional stability throughout the simulation, choosing the parameters of the PBDM so as to be physically accurate might not be a trivial procedure [35]. The optimisation of such parameters using the component-based software framework is the topic of Chapter 5.

### 2.3.3 Accuracy

The quantification of accuracy is explored through the procedure of verification and validation (V&V). Whilst ultimately dependent on the intended use of the application, a definition of V&V is given in [36] as:
1. “Verification: the process of determining that a model implementation accurately represents - in accordance to the developer - the conceptual description of the model and the solution to the model.”

2. “Validation: the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model.”

Verification might be viewed as the successful implementation of the model on a computer, whilst validation is whether or not the computational solutions from the model hold true to its real-life or analytical counterparts. Indeed, “the ultimate test of accuracy is how well the numerical method reproduces known results”[37]. However, whilst real data exists for some select problems at a select few time instances [38], most problems have no real data of entire deformation histories with which to compare.

In engineering disciplines, accuracy holds the most importance and so the PBDMs used provide accurate numerical approximations to the solutions of the equations of motion (found in Section 2.4). The FEM [39] is often used to solve the nonlinear solid mechanics equations on a mesh at a sufficiently high resolution. This is widely regarded as being the most accurate PBDM [3][16][40] and is the topic of Section 2.5.1. Zienkiewicz and Zhu [41] introduced the Z2 error estimator, which is an a posteriori error estimate that aims to reduce discretisation error. The determination of a sufficiently high resolution can be determined by using their provided error. The solution might be rapidly changing between neighbouring discretised values, which implies that more discrete points are needed to better represent the rapid change in the solution. In other words, the estimated error is deemed too large in this particular region on the discrete mesh and so the mesh should be more finely refined in this area [42]. See Figure 2.5. If we fail to do
2.3. MODELLING DEFORMATIONS IN COMPUTER GRAPHICS

Figure 2.5: The modelled geometric body is a parallelepiped and is discretised using cubic elements. On the right, the discretisation of the body is refined and constructed of more cubic elements. This procedure reduces discretisation error in areas where the numerical solution is rapidly changing. Refinement strategies such as this are not computationally trivial procedures and are not suitable for real-time interactive simulations. They also introduce more discrete nodal points at which to solve the equations, which (also) reduces computational performance.

This then fine detail of the deformation will be lost. Once we obtain a solution of minimal discretisation error - corresponding to a sufficiently finely discretised mesh - we term the NLFEM a *gold standard* for accuracy.

Due to the intensive computational effort required to achieve such accuracy with PBDMs such as NLFEM, however, they are not suitable for real-time CG applications. Thus, *surrogate* PBDMs must instead be used to approximate the solutions produced by a gold standard PBDM in order to obtain real-time computational performance. The accuracy of such surrogate PBDMs typically is evaluated only qualitatively through visual plausibility [2][43] or, rarely, quantitatively. However, such quantitative validation is typically only performed at a single instance in time, as mentioned in Section 1.3. A solution to the problem of quantitatively validating an entire deformation history of a PBDM is presented in Chapter 3.

The next Section introduces the equations of motions to be numerically solved by the PBDMs presented later on in this Chapter. Whilst all PBDMs aim to provide numerical solutions to the equations of motion, each PBDM models the internal force and the measure of deformation differently and so produces a different deformation. In other words, each leverages computational performance,
accuracy and stability differently to deform the modelled body. We present a means to measure the resulting difference in accuracy in Chapter 3.

2.4 Equations of motion

This Section presents the key equations of motion used to deform physical bodies. The equations describe how a body internally reacts to external loads and their solution produces a displacement field across the body. PBDMs model both the measure of deformation and the internal force differently to give differing displacement fields. The PBDMs that are used to model deformations are used as examples in presenting the contributions of Chapters 3, 5 and 6.

2.4.1 Lagrangian framework

The motion of a particle as described by Newton obeys Newton’s second law (2.1). The motion of a continuum can be obtained through the (energy) formulation of the resulting mechanical system as an partial differential equation in terms of the potential energy of the system, $V$, and kinetic energy, $T$:

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{v}(t)} \right) - \frac{\partial L}{\partial u(t)} = f_{\text{ext}}(t),$$  

where $u(t)$ and $v(t)$ are the displacement and velocity of a particle, respectively. In this equation, $L = T - V$ is known as the Lagrangian of the system and $f_{\text{ext}}$ is the applied external loads. A key observation of this framework is that the dynamic behaviour of a mechanical system is fully determined by functions of the potential and kinetic energies.

Whilst Newton’s description of mechanics requires us to use a Euclidean (Cartesian) coordinates system, the Lagrangian framework is usable in different,
more appropriate coordinate systems. However, for numerically solving equations on a computer, Cartesian coordinates are used because they require fewer computations. It is often the case that the Lagrangian framework is easier to work with because it requires a statement of the (scalar) potential energy field rather than a force (vector) field [44]. Some PBDMs (such as mass-spring systems, introduced in Section 2.5.5) form the potential energy fields in the form of simple constraints that must be satisfied at each step. This is easily achievable when using the Lagrangian framework and the numerical procedure required to satisfy each constraint is not an intensive one.

The kinetic energy describes the energy as a result of the motion of the system and for an elastic continuum $\Omega$ with volume $V_t$ at time $t$ is:

$$T(v(t)) = \int_{\Omega_t} \rho(x(t))||v(x(t), t)||^2_2 dV_t,$$  \hspace{1cm} (2.3)

where $\rho(x)$ is the current volumetric mass density field of the body (its mass per unit volume) and $|| \cdot ||_2$ is the Euclidean norm. It is noted that the expression for kinetic energy is independent of the current displacements $u$ and time $t$. The potential energy describes the energy of the interaction between particles inside the body. For conservative systems, $V = V(u)$ only and is independent of $v$ and $t$:

$$V(u) = V(X) = \int_{\Omega_t} W(\hat{X}, \hat{x}) dV_t,$$  \hspace{1cm} (2.4)

where $W$ is the elastic potential energy density function at $\hat{x}$ under the displacement to $\hat{X} = \hat{x} + \hat{u}$ and $\hat{x}$ in the integrand represents the value of $x$ throughout the volume $\Omega_t$. 
CHAPTER 2. BACKGROUND

2.4.2 Spatial discretisation

A discretisation of the elastic body into discrete nodal points allows us to calculate numerical solutions to equation (2.2) at these nodal points on a digital computer. For a discretisation into \( n \) nodes, explicit Lagrangian methods *lump* the mass to the nodes so that the total mass of the body is split between the nodal points. This can be represented mathematically by the *mass matrix* \( M(t) \in \mathbb{R}^{3n \times 3n} \) that is, in general, a diagonal matrix with diagonal entries given by \( m_i \), the mass proportion at node \( i \):

\[
(m_1, m_1, m_2, m_2, \ldots, m_n, m_n, m_n).
\]

We can write the kinetic energy equation (2.3) in terms of these discrete points as:

\[
T = T(v(t)) = \frac{1}{2} v^T(t)M(t)v(t).
\]

The total mass \( m(t) \) of a continuous body is found by integrating over the entire volume \( V_t \). \( m \) is dependent on the density field \( \rho(x) \) and the volume of the body:

\[
m = \int_{\Omega_t} \rho(t)dV_t.
\]

The mass matrix \( M(t) \) modelling the body numerically approximates the distribution of the total mass \( m \), for example:

\[
m(t) = \sum_{i=1}^{n} m_i(t).
\]

Often the time dependency of \( M \) is removed so that \( M(t) = M \) and the mass (matrix) is constant. For the lumped mass matrix approach, the substitution of
the above discretised expression of the kinetic energy into equation (2.2) gives:

\[ M\ddot{u}(t) + f_{\text{int}}(u(t), t) = f_{\text{ext}}(t), \quad (2.5) \]

where

\[ \ddot{u}(t) = \frac{d^2 u(t)}{dt^2} = a(t) \]

is the nodal acceleration and

\[ f_{\text{int}}(u(t), t) = -\frac{\partial V(u(t))}{\partial u(t)} \quad (2.6) \]

are the internal forces at each discrete node at time \( t \). Note that the negative sign implies that the potential energy is equal to the work you must do against \( f_{\text{int}} \) in order to displace a material point.

With equation (2.5) we have arrived (back) at Newton’s description of mechanics from the Lagrangian description anyway. In equation (2.5), the equations of motion are expressed as a balance between externally applied loads, \( f_{\text{ext}}(t) \), and the sum of the d’Alembert inertial forces [39] and the internal forces, \( M\ddot{u}(t) + f_{\text{int}}(u(t), t) \). For a steady state problem, there are no inertial terms and so (2.5) reduces to

\[ f_{\text{int}}(u(t), t) = f_{\text{ext}}(t). \quad (2.7) \]

To describe the relative motion of material points on a continuum, the local deformation at any point must be quantified. For the interested reader, details of the theory that follows is found in [5][6].
2.4.3 Deformation gradient tensor

If the directional vector of a line element in the undeformed body, originating at \( x \), is given as \( dx \) then its end is \( x + dx \). The corresponding line element in the deformed body is \( dX = X(x + dx) - X(x) \). Substituting in the Taylor expansion of \( X(x + dx) \), assuming \( |dx| << 1 \):

\[
dX \approx X(x) + \frac{\partial X}{\partial x}(x) \cdot dx - X(x) = \frac{\partial X}{\partial x}(x) \cdot dx \equiv F(x) \cdot dx.
\]  

(2.8)

\( F \) represents the material deformation gradient tensor and describes the (linear) mapping of undeformed line elements to deformed line elements.

2.4.4 Strain

Strain evaluates how much a given displacement differs (locally) from a rigid body displacement; if any relative stretching or rotation has occurred then there is some strain. The deformation gradient tensor, \( F \), maps material lines in the undeformed configuration to the deformed configuration. The squares of line lengths are always positive, thus the difference of square lengths of the same material line between undeformed and deformed configurations would provide an objective measure of strain. The difference in square lengths is given by

\[
dX \cdot dX - dx \cdot dx \approx F(x) \cdot dx \cdot F(x) \cdot dx - dx \cdot dx = (F^T F - I)dx \cdot dx.
\]

by (2.8), where \( c = F^T F \) is the (right) Cauchy-Green deformation tensor and \( I \) is the identity matrix. An objective measure of how much the material has strained
2.4. EQUATIONS OF MOTION

at any given point is given by:

\[ e = \frac{1}{2}(F^T F - I) = \frac{1}{2}(c - I), \] (2.9)

where \( e \) is the Green-Lagrange strain tensor in a Cartesian coordinate system.

Deformation and Displacement

The displacement of a material point from its initial position is

\[ u(x, t) = X(x, t) - x \] (2.10)

and so

\[ \frac{\partial u(x, t)}{\partial x} = \frac{\partial X(x, t)}{\partial x} - \frac{\partial x}{\partial x}, \]

which is

\[ \nabla_x u = \nabla_x X - I = F - I. \] (2.11)

\( \nabla \) is the gradient operator and \( \nabla_x u \) is called the material displacement gradient tensor and describes the amount that a material point has displaced upon deformation (but not as a result of a rigid body translation).

Substituting (2.11) into the definition of the Green-Lagrange strain tensor, (2.9), yields

\[ e = \frac{1}{2}(\nabla_x u + I)^T(\nabla_x u + I) - I). \]

\[ = \frac{1}{2}(\nabla_x u + \nabla_x u^T + \nabla_x u^T \nabla_x u), \] (2.12)

giving the measure of strain as a function of the displacement gradients.
CHAPTER 2. BACKGROUND

2.4.5 Hyperelastic materials

A hyperelastic material is one for which the strain energy density function, $W$, is a function of the deformation gradient tensor only:

$$W = W(F) = W(\nabla_x X(x)).$$  \hspace{1cm} (2.13)

The work done by internal stresses per unit undeformed volume of the body—called the stress-power—is such that

$$\frac{dW}{dt} = \sigma : \frac{de}{dt}.$$

Using the chain rule for differentiation, the stress can be written as:

$$\sigma = \frac{\partial W}{\partial e}.$$ \hspace{1cm} (2.14)

From the Euler-Lagrange equations of motion (2.5), the balance of linear momentum for a hyperelastic material in the undeformed body configuration is:

$$\rho_0 \frac{\partial \mathbf{v}}{\partial t} = \nabla \cdot \sigma + f_{\text{ext}},$$ \hspace{1cm} (2.15)

where the divergence operator $\nabla \cdot$ acts across the rows of the stress tensor, $\sigma$, when the latter is expressed as a (symmetric) matrix:

$$\sigma = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{pmatrix}.$$
Principle of Virtual Displacements

The governing equations of hyperelasticity (2.15) can instead be expressed as a variational principle to solve for unknown nodal displacements [39], called the principle of virtual work. From this, the principle of virtual displacements (PVD) is obtained. The solution is still a displacement field. This is the method used in NLFEM in Section 2.5.1.

Whilst the linear momentum is in balance, the body is subject to an instantaneous, infinitesimal virtual displacement field $\delta \mathbf{u}$. The virtual displacement field is arbitrary and unrelated to the real displacement field. It must be 0 on the boundary where real displacement boundary conditions are prescribed. Let $\delta \mathbf{e}$ be an arbitrary virtual strain field, associated to the virtual displacement field $\delta \mathbf{u}$, unrelated to the real strain field. Note that there is no physical meaning to the virtual displacements.

As summarised in [39], the external virtual work is the real net body forces in the undeformed configuration, $\mathbf{f}$, and real surface forces in the deformed configuration, $\mathbf{T}$, multiplied by the virtual displacements, integrated over the undeformed volume and deformed surface of the body, respectively:

$$\delta W_e = \int_{\Omega_0} \rho_0 \left( \mathbf{f} - \frac{d \mathbf{v}}{d t} \right) \cdot \delta \mathbf{u} \, dV_0 + \int_{\partial \Omega_0} \mathbf{T} \cdot \delta \mathbf{u} dS_t. \quad \text{(2.16)}$$

The internal virtual work is due to the work done due to the virtual strains:

$$\delta W_i = \int_{\Omega_0} \mathbf{\sigma} : \delta \mathbf{e} \, dV_0. \quad \text{(2.17)}$$

Equation (2.17) is the internal virtual work and is the real stresses multiplied by the virtual strains, integrated over the volume of the body. Note that there is no specified constitutive law, as discussed below.
The real stresses correspond to the real forces and the virtual strains correspond to the virtual displacements, which satisfy the real displacement boundary conditions. The principle of virtual work is then obtained by equating (2.16) and (2.17):

$$\int_{\Omega} \rho_0 \left( \mathbf{f} - \frac{\partial \mathbf{u}}{\partial t} \right) \cdot \delta \mathbf{u} \, dV_0 + \int_{\partial \Omega_t} \mathbf{T} \cdot \delta \mathbf{u} \, dS_t = \int_{\Omega_0} \sigma : \delta \mathbf{e} \, dV_0. \quad (2.18)$$

The principle of virtual displacements is then the restatement of the principle of virtual work, (2.18), but with the virtual strain field represented in terms of the virtual displacement field:

$$\int_{\Omega} \rho_0 \left( \mathbf{f} - \frac{\partial \mathbf{u}}{\partial t} \right) \cdot \delta \mathbf{u} \, dV_0 + \int_{\partial \Omega_t} \mathbf{T} \cdot \delta \mathbf{u} \, dS_t = \int_{\Omega_0} \sigma : \nabla_x \delta \mathbf{u} \, dV_0. \quad (2.19)$$

Constitutive Laws

The equations of motion must be augmented by a constitutive law that determines the stress as a function of the deformation of the body via (2.14). Popular strain energy functions are [45]

- **Neo-Hookean**

  $$\mathcal{W} = C_1 (I_1 - 3) + c I_3 - d \ln \left( \sqrt{I_3} \right),$$

  where $C_1, c, d$ are constants.

- **St. Venant-Kirchhoff**

  $$\mathcal{W} = \frac{\lambda}{2} \text{tr}(\mathbf{e})^2 + \mu ||\mathbf{e}||_F^2 = \frac{\lambda}{8} (I_1 - 3)^2 + \frac{\mu}{4} \left[ (I_1 - 3)^2 + 4(I_1 - 3) - 2(I_2 - 3) \right],$$

  where $||A||_F = \sqrt{\text{trace}(A^T A)}$ is the Frobenius norm of the real matrix $A$.
The $I_i$ are the strain invariants:

\begin{align*}
I_1 &= \text{tr}(F^T F) = \mu_1 + \mu_2 + \mu_3 \\
I_2 &= \frac{1}{2} \left( (\text{tr}(F^T F))^2 - \text{tr}((F^T F)^2) \right) = \mu_1 \mu_2 + \mu_1 \mu_3 + \mu_2 \mu_3 \\
I_3 &= \det(F^T F) = \mu_1 \mu_2 \mu_3.
\end{align*}

(2.20)

The eigenvalues of $F^T F$ are $\mu_i$. Their square roots are the principal stretches of the deformation, in the direction of the principal axes of stretch, given by the associated eigenvectors $\hat{v}_i$. Both strain energy functions are used in the gold standard NLFEM model in Chapter 3.

**Motivation for alternative PBDMs**

The method of modelling the internal forces, $f_{\text{int}}$, is essentially what differentiates PBDMs. Typically, $f_{\text{int}}$ is nonlinear and its numerical computation is a computationally intensive procedure. Traditionally, approximations are made to the calculation of $f_{\text{int}}$ that reduce the complexity of the calculations, at the cost of a loss in accuracy of potential deformations. This thesis presents a means of quantifying this loss over an entire deformation history. Next, the most commonly used PBDMs in the literature to numerically solve (2.5) are introduced.

### 2.5 Simulation methods of solid bodies

PBDMs attempt to solve the Euler-Lagrange equations of motion (2.5) for the displacement values at each node $i$, $u_i$, at the next time step. Each PBDM models and computes both the amount of strain and the internal forces, $f_{\text{int}}$, differently.
and this modelling and computation is what differentiates each PBDM. This Section introduces variations of the FEM, followed by mass-spring systems (MSS) and then position-based dynamics (PBD) models, which are the most common PBDMs found in the real-time CG literature. Many other PBDMs exist, which are periodically summarised in state-of-the-art-reports reports such as by Nealen et al. [2], Zhang et al. [28] and Huang et al. [46]. This aim of this Section is to explain how these alternative model formulations link to the general physical formulation presented in Section 2.4.

2.5.1 Nonlinear finite element methods (NLFEM)

For a detailed introduction to the finite element method used in this context, the reader is referred to [39]. We wish to determine the displacement field at each update step of the simulation. To do this, we work in the Lagrangian framework and maintain knowledge of the undeformed configuration of the nodes. This method models both the internal forces and deformation measure (the strain) nonlinearly, which requires more computations and therefore makes it unsuitable for real-time CG applications when the mesh is finely discretised. The following theory shows how we find the unknown displacement field using the NLFEM. Note that we use the Einstein summation convention; we sum over repeated indices in any given expression.

The body is discretised into smaller finite elements whose union forms the entire body. See Figure 2.6. In Galerkin’s method [39] the undeformed positions are interpolated between by choosing \( n \) trial shape functions \( \phi_1(x), \phi_2(x), \ldots, \phi_n(x) \) so that:

\[
x_i = \sum_{l=1}^{n} x_i^{(l)} \phi_l,
\]

where \( x_i^{(l)} \) is the \( i \)-th coordinate of the undeformed position of the \( l \)-th node. The
shape functions provide a piecewise interpolation between the undeformed positions of the nodes. In the Lagrangian framework, we follow material points over time so that the deformed positions (or, equivalently, the displacements) of the nodes are relative to the undeformed positions of the nodes. We use an isoparametric method, which means that the same trial shape functions also interpolate the displacements, $\mathbf{u}$:

$$ u_i = \sum_{l=1}^{n} u_i^{(l)} \phi_l, $$

Since the basis functions are fixed in space, the derivatives are given by:

$$ \frac{\partial u_i}{\partial x_j} = \sum_{l=1}^{n} u_i^{(l)} \frac{\partial \phi_l}{\partial x_j}. $$

The upshot of this is that in the principle of virtual displacements (PVD), given in equation (2.19), the variations in displacements are due to variations in the
discrete nodal displacements only and not the shape functions:

\[ \delta u_i = \sum_{l=1}^{n} \delta u_i^{(l)} \phi_l, \]
\[ \delta \frac{\partial u_i}{\partial x_j} = \sum_{l=1}^{n} \delta u_i^{(l)} \frac{\partial \phi_l}{\partial x_j}. \]

Using these expression in the PVD we can write:

\[ \sum_{l=1}^{n} \int_{\Omega_0} \left[ \sigma_{ij} \frac{\partial u_k}{\partial x_i} \frac{\partial \phi_l}{\partial x_j} - \rho_0 \left( f_k - \frac{\partial^2 u_k}{\partial t^2} \right) \phi_l \right] \delta u_k^{(l)} dV_0 - \int_{\partial \Omega_t} T_k \delta u_k^{(l)} \phi_l = 0 \]
\[ \sum_{l=1}^{n} \left( \int_{\Omega_0} \left[ \sigma_{ij} \frac{\partial u_k}{\partial x_i} \frac{\partial \phi_l}{\partial x_j} - \rho_0 \left( f_k - \frac{\partial^2 u_k}{\partial t^2} \right) \phi_l \right] dV_0 - \int_{\partial \Omega_t} T_k \phi_l \right) \delta u_k^{(l)} = 0. \]

Since the variations of all nodal positions are arbitrary, the terms in large outer parentheses must vanish independently, which gives one discrete equation for each coordinate at each node in this element:

\[ b_{kl} = \int_{\Omega_0} \left[ \sigma_{ij} \frac{\partial u_k}{\partial x_i} \frac{\partial \phi_l}{\partial x_j} - \rho_0 \left( f_k - \frac{\partial^2 u_k}{\partial t^2} \right) \phi_l \right] dV_0 - \int_{\partial \Omega_t} T_k \phi_l. \quad (2.21) \]

The value \( b_{kl} \) will have contributions from all elements that contain the node with the local node index \( l \). This is known as the element assembly procedure. Separate surface finite elements handle the external loads arising from surface tractions due to surface pressures or collisions on \( \partial \Omega_t \), which are common with user interactions within IVEs. The assemblage of each \( b_{kl} \) leads to the so-called residual vector \( b \). Note that the element assembly can be performed in parallel since the PVD is evaluated independently for each element; i.e. we have an embarrassingly parallel problem. Each element contributes to the resulting residual vector independently and we obtain a vector reduction problem in the context of (task) parallel programming.
The unknown displacements $\mathbf{u}_{t+h}$ that solve the equations (2.21) at the current time step can be obtained numerically by using the Jacobian, $J$:

$$-J \delta \mathbf{u} = - \frac{\partial \mathbf{b}}{\partial \mathbf{u}_t} \delta \mathbf{u} = \mathbf{b}. \quad (2.22)$$

From which we have $\mathbf{u}_{t+h} = \mathbf{u}_t + \delta \mathbf{u}$. In general (2.21) is nonlinear, which comes from the following two modelling assumptions.

- Geometric nonlinearity: the Green-Lagrange strain tensor contains terms nonlinear in the displacement gradients (Section 2.4.4).

- Material nonlinearity: the stress-strain constitutive law is nonlinear (Section 2.4.5).

This means that the solution of (2.21) must be obtained iteratively using equation (2.22) until some tolerance on the size of the residual vector $\mathbf{b}$ is met. This procedure is outlined in Algorithm 1.

**Summary**

The NLFEM models nonlinear material response to external loads using equations from continuum mechanics that describe the motion of solid bodies. The measure of deformation used is the Green-Lagrange strain tensor which measures the change in square lengths of line elements from the undeformed to deformed body configurations. The material response to such strains can be modelled nonlinearly using a constitutive law as described in Section 2.4.5. The effect of a nonlinear strain measure and a nonlinear constitutive law is that the produced displacement field under external loads accurately models real life phenomena. The solutions of any NLFEM experiments conducted in this thesis are obtained using *oomph-lib* of Heil and Hazel, which is an open source, object oriented multi-physics FEM.
Algorithm 1 The solution of the displacement field is obtained iteratively for problems which are nonlinear and cannot be solved in one (linear) iterative step. If the system is linear then the system is solved in one step. The evaluation of the size of the residual vector $b$ in step 7 is commonly taken as the maximum component of $b$ or the Euclidean distance of $b$ (from 0). The linear system solve in step 9 is typically performed using either direct methods using some kind of matrix decomposition (e.g. LU decomposition) or iterative methods such as the conjugate gradient method [9]. In explicit numerical integration schemes, the matrix $J$ reduces to the so-called mass-matrix, $M$. In this case there are no inter-dependencies between elements in $J$. Often the mass-matrix is diagonal, in which case we can find $δu$ directly by inverting $M$.

1: procedure NONLINEAR SYSTEM SOLVE FOR THE DISPLACEMENT FIELD USING NEWTON-RAPHSON
2: Set the maximum iteration count, $n$.
3: Set the iteration count, $i ← 0$.
4: Set the tolerance, tol.
5: Calculate the residual vector $b$ from PVD in equation (2.21) using $u_t$
6: Set the initial guess $u_{t+h} ← u_t ↪ The previous $u$ are a good initial guess
7: while Size($b$) > tol AND $i < n$ do
8: Calculate the Jacobian: $J = \frac{∂b}{∂u_{t+h}}$
9: Solve $-Jδu = b$ for $δu$
10: $u_{t+h}^{i+1} ← u_{t+h}^{i} + δu$
11: Recalculate $b$ using $u_{t+h}^{i+1}$
12: $u_{t+h} ← u_{t+h}^{i}$
13: return $u_{t+h}$
library that calculates solutions of physical scenarios. In this way, the software serves as a black box for us to obtain gold standard, reference deformation data to compare against. For any scenario, the user can specify material properties $E$ and $\nu$ which then are converted by oomph-lib into the constants present in the strain energy functions in Section 2.4.5.

**Drawbacks** of NLFEM are the performance bottlenecks of the method that come from the following.

- The Newton-Raphson algorithm requires numerous iterations, because the problem is nonlinear. The following three numerically intensive procedures must be performed *at each iteration*.

- The element-by-element assemblage of the residual in line 11 of Algorithm 1. This is a numerical intensive procedure because of:

  - calculating the nonlinear Green-Lagrange strain, which can be nonlinear in the displacement gradients;
  
  - determining the stress from the constitutive law, which can have a nonlinear relationship to the strain measure.

- The element-by-element assemblage of the Jacobian in line 8 of Algorithm 1, obtainable using finite differences, say.

- Solving the resulting linear system (in line 9 of Algorithm 1). The system scales linearly with the number of unknown displacement values, so for finely discretised meshes with many unknown values, the linear system solve is a big task.
2.5.2 Linear finite element methods (LFEM)

The LFEM follows the solution procedure of NLFEM and the PVD can be used to find the unknown nodal displacements. Both a linearised measure of strain and a linear constitutive law relating stress and strain are used to obtain the stress at each material point at each update step, from which the internal force at each node is obtained. If either of these are nonlinear, then the method is NLFEM. The linear approximation holds under the crucial assumption that any strains (and therefore deformations) will be small. A linear stress-strain relationship implies that:

- the deformation behaviour of the hyperelastic material due to internal body stresses scales linearly with the amount of strain that is produced;

- the deformation is reversible.

It is the linearisation of both the strain measure and the constitutive law that forms the LFEM. This provides a big computational saving compared to NLFEM because:

- the constitutive law linking nodal displacement gradients to (nodal) internal stresses can take the form of a constant matrix; the behaviour scales linearly with the measured nodal displacements. This is a generalisation of Hooke’s law to 3D solid bodies. This is unlike NLFEM, whose material response depends on the current displacement field and does not scale linearly. We can use a constant matrix because of the modelling assumption that any material point displacement will be small. The measured displacements are connected to the produced internal forces at each node by a stiffness matrix, which is constant.
• the resulting linear system converges on the unknown displacements in one step in Algorithm 1.

The linearised strain measure is discussed below.

**Cauchy strain**

The Green-Lagrange strain tensor (2.12) generally contains non-negligible nonlinear terms. Under the assumption that any material point displacements will be small, however, the nonlinear displacement gradient terms of (2.12) can be truncated, giving the Cauchy strain tensor:

\[ \varepsilon = \frac{1}{2}(\nabla_x u + \nabla_x u^T). \]  

(2.23)

The violation of the assumption of small displacements that motivates the truncation of the higher order terms is what gives the visual artifacts observable in the examples given in Figures 1.3 and 2.7.

Alternatively to using PVD, the *explicit finite element method* utilises the so-called “lumped” mass matrix as mentioned in Section 2.4.2, which splits the total mass of the solid body over all the nodes. The explicit finite element method can be thought of as having the nodes inside a single element connected as a network of *hypersprings* [2]; the internal force at each node of an element is then dependent on the displacement of every node in that same element.

**Stiffness matrix**

We mentioned that the means of obtaining the internal forces at each node from the measured displacements can be made using the so-called stiffness matrix. The stiffness matrix, denoted K, is constant because it is linearised around the
undeformed configuration:

\[
\begin{align*}
\mathbf{f}_{\text{int}}(\mathbf{x} + \mathbf{u}) &= \mathbf{f}_{\text{int}}(\mathbf{x}) + \frac{\partial \mathbf{f}_{\text{int}}}{\partial \mathbf{x}}(\mathbf{u}) + O(|\mathbf{u}|^2) \\
&= \mathbf{f}_{\text{int}}(\mathbf{x}) + K|\mathbf{x}|\mathbf{u} + O(|\mathbf{u}|^2) \\
&\approx \mathbf{f}_{\text{int}}(\mathbf{x}) + K|\mathbf{x}|\mathbf{u},
\end{align*}
\]

since \( \mathbf{u} \) is assumed to be small. The stiffness matrix models the force response of each node in the discretised body in response to the nodal displacements of other nodes. Subsequently, it has many non-zero entries matrix because each indexed row (and, therefore, column) of the matrix corresponds to the connectivity of each indexed node with other nodes via its shared finite elements. The matrix is sparse, however, since the matrix contains a larger proportion of zero element entries as the number of degrees of freedom (the unknown displacement values) of the system increases. Sparse matrices are advantageous because we do not need to store the many zero entries, thus leaving a smaller memory footprint. The linear explicit finite element method (LFEM) solves equation (2.5) for each node and models the force \( \mathbf{f}_{\text{int}} \) at each node as:

\[
\mathbf{f}_{\text{int}} = \nabla \cdot \mathbf{\sigma}
\]

from equation (2.15). Numerically speaking, this divergence calculation of the stress is made using the stiffness matrix multiplied with the measured nodal displacements:

\[
\mathbf{f}_{\text{int}} = K\mathbf{u}.
\]

The stiffness matrix \( K \) can be decomposed into three steps that link

- displacements → strains → stresses → internal forces
in a process that is called the stiffness matrix assembly. The stiffness matrix assembly of LFEM models is presented in Appendix A.1.

Summary

LFEM models are similar to NLFEM models but with the modelling assumption that any deformations will be small. These assumptions are in the form of using the linear Cauchy strain measure and the linear constitutive law. The benefit to this is that fewer numerical computations are required to compute the internal forces at each update step for each node and the Newton-Raphson method of Algorithm 1 converges in one step because the equations are linearised. The stiffness matrix used to compute the internal nodal forces at each update step is constant and so can be computed once at the start of the simulation and stored for later update steps, providing a large computational saving.

Drawbacks If we store the stiffness matrix then this can take up substantial memory since the matrix size scales quadratically with an increase in nodal count. The LFEM model is only valid for small displacements and if this modelling assumption is broken then we get visual artifacts relating to volume conservation. We will see examples of this in Section 2.5.4.

2.5.3 Corotational finite element methods

The visual artifacts that are present from LFEM under large deformations are due to both large stretches (equivalent to material points becoming displaced further apart) and large relative rotations of material points. The Cauchy strain tensor is not invariant under rigid body rotation: i.e. the same strain is not achieved under a (further) rotation of the entire body in space. As an example, a body is under no strain and is rotated in the $xy$ plane by an angle $\theta = \frac{\pi}{2}$ radians. The
CHAPTER 2. BACKGROUND

deflection gradient tensor is then

\[
F = \begin{bmatrix}
\cos \theta & \sin \theta & 0 \\
-\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{bmatrix} = \begin{bmatrix}
0 & 1 & 0 \\
-1 & 0 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

so that

\[
\begin{bmatrix}
X_1 \\
X_2 \\
X_3
\end{bmatrix} = F \begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} = \begin{bmatrix}
-x_2 \\
x_1 \\
x_3
\end{bmatrix}.
\]

The displacement is

\[
\begin{bmatrix}
u_1 \\
u_2 \\
u_3
\end{bmatrix} = \begin{bmatrix}
X_1 \\
X_2 \\
X_3
\end{bmatrix} - \begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} = \begin{bmatrix}
-x_1 - x_2 \\
-x_1 - x_2 \\
0
\end{bmatrix}
\]

and the displacement gradient is

\[
\nabla_x u = \begin{bmatrix}
-1 & -1 & 0 \\
1 & -1 & 0 \\
0 & 0 & 0
\end{bmatrix}.
\]

Finally, the Cauchy strain is

\[
\varepsilon = \frac{1}{2}(\nabla_x u + (\nabla_x u)^T) = \begin{bmatrix}
-1 & 0 & 0 \\
0 & -1 & 0 \\
0 & 0 & 0
\end{bmatrix} \neq 0_3,
\]

so that strain is measured despite no deformation.

One solution to have a rigid body rotation invariant strain measure is to use
the Green-Lagrange strain. Using the previous example,

\[
e = \frac{1}{2} \left( \nabla_x u + (\nabla_x u)^T + \nabla_x u (\nabla_x u)^T \right)
\]

\[
= \frac{1}{2} \left( \begin{bmatrix}
-1 & -1 & 0 \\
1 & -1 & 0 \\
0 & 0 & 0 
\end{bmatrix} + \begin{bmatrix}
-1 & 1 & 0 \\
-1 & -1 & 0 \\
0 & 0 & 0 
\end{bmatrix} + \begin{bmatrix}
2 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 0 
\end{bmatrix} \right) = 0_3,
\]

so that the Green-Lagrange strain tensor is shown to invariant under rigid body rotation. However, this makes the stress-displacement relationship nonlinear because of the nonlinear displacement gradient terms. Alternatively, the rotational parts of the displacements can be extracted, which rotates the nodes and elements back to their initial orientation [11]. The remaining displacements are then due to stretches only. The internal force computation of (2.5) now becomes:

\[
f_{\text{int}} = R_e K_e (R_e^{-1} X - x),
\]  

(2.25)

where \( R_e \) is the rotation matrix of the element from its initial orientation and \( K_e \) is the element stiffness matrix as in LFEM. Corotational FEM still exhibits visual artifacts for large stretches, however, but is computationally more efficient than NLFEM.

**Summary**

The visual artifacts that result from using the linear Cauchy strain measure are due to large stretches and large relative rotations. CoRotFEM models avoid those due to rotations by rotating the deformed body back to the reference (undeformed) configuration, measuring the Cauchy strain (and internal forces using the stiffness matrix) and then rotating the body back to the deformed configuration.
Drawbacks The drawbacks of CoRotFEM models lie in the linear constitutive law problem that is also present in LFEM; the amount of stress still scales linearly with the amount of measured strain. In addition, there is more performance overhead due to the additional element-wise rotations for each element at each update step in order to obtain the internal forces. We find this not to be a problem in practice, however, since the method still converges linearly in the Newton-Raphson algorithm. Finally, the element stiffness matrices can be stored for use in equation (2.25) to save computation time at the expense of memory overheads.

2.5.4 Comparison of FEM

The NLFEM, LFEM and CoRotFEM models presented in this Section all have the same solution procedure but exhibit different modelling assumptions. These assumptions relate to the nature of the deformation that is being modelled. The drawbacks of the LFEM and CoRotFEM models are visualised in Figure 2.7, in which the three bodies are fixed at one end (vertically in one case and horizontally in a second) and released under gravity. In the NLFEM model (left, red) we can model many complex nonlinear deformations without visual artifacts. In the LFEM model we assume that any deformations that occur are so small that a linear approximation of the deformation measure (i.e. the strain) is approximately equivalent to the nonlinear deformation measure. This includes the amount of stretch between material points and also the relative rotation of material points. If this assumption is violated, then the volume is not conserved. We can see this in Figure 2.7. In the right image, the LFEM model (centre, blue) exhibits large relative rotational deformations and grows in volume. The CoRotFEM model (right, orange) is rotationally invariant and therefore does not exhibit
2.5. SIMULATION METHODS OF SOLID BODIES

Figure 2.7: We model two cantilever beam scenarios under gravity. The material properties are $E = 4000 \text{Nm}^{-2}$, $\nu = \frac{1}{4}$. In the left scenario the beams are fixed along the top surface and in the right scenario the beams are fixed along the back surface. The visual artifacts of the LFEM (centre, blue) and CoRotFEM (right, orange) models are depicted. This arises because we violate the linear modelling assumptions of the deformations. The NLFEM (left, red) model does not exhibit these artifacts because it also contains the nonlinear displacement gradient terms in its (Green-Lagrange) strain measure.

this volumetric growth. When we model the stress-strain relationship of the material with a linear constitutive law then we assume that the amount of stress scales linearly with the amount of strain. In the left image we see the bodies stretch under the gravitational body force. The deformation measure of the NLFEM model includes nonlinear terms and so measures a larger strain than the LFEM and CoRotFEM models as the body gets stretched in the downward direction (and compressed in the other two Cartesian directions). We can see the effect of ignoring the nonlinear terms in the LFEM and CoRotFEM models since the bodies grow in volume. The smaller measures of strain produce smaller internal stresses, which produces a smaller internal restoring force and makes
the body stretchier. We find in general that for the deformations considered in the CG literature, relative rotational deformations are more common than overly stretchy deformations, which makes CoRotFEM models appropriate choices for the deformations considered because they are rotationally invariant.

As a final note, in this thesis we do not focus on runtime performance of the PBDMs that we use, but rather focus entirely on making quantitative comparisons. We leave performance considerations to future work.

2.5.5 Mass-spring systems (MSS)

The potential energy field in the Lagrangian framework of equation (2.2) describes the energy of the interaction between discrete particles. For a mass-spring system, particles interact along springs that lie in straight lines between the particles, or point masses. A simple example is illustrated in Figure 2.8. These springs are linear stretch constraints that obey Hooke’s law and keep the nodes from displacing too far apart from each other. The springs are mass-less. A spring $s_{ij}$ of stiffness $k$ connecting two masses $i, j$ at positions $x_i$ and $x_j$ has a rest length $||x_j - x_i|| = l_s$ when the system is in equilibrium. The potential energy field is

![Figure 2.8: This illustrated mass-spring system connects point masses $m_1, m_2, m_3$ by two springs with stiffness values $k_1, k_2$ that constrain the motion of the point masses under the influence of any applied external loads.](image)
2.5. SIMULATION METHODS OF SOLID BODIES

defined as:

\[ V = \frac{1}{2} k (||X_j - X_i|| - l_s)^2 = \frac{1}{2} k (||u_j - u_i|| - l_s)^2. \]

We arrive at Hooke’s law by the equation (2.6):

\[
\begin{align*}
    f_{\text{int}}^{s_{ij}}(x_i) &= -\nabla u_i V = k \delta u_{ij}, \\
    f_{\text{int}}^{s_{ij}}(x_j) &= -\nabla u_j V = -k \delta u_{ij}. 
\end{align*}
\]  

(2.26)

where \( f_{\text{int}}^{s_{ij}} \) is the internal spring force of \( s_{ij} \) and

\[ \delta u_{ij} = \frac{u_j - u_i}{||u_j - u_i||} (||u_j - u_i|| - l_s) \]

is a measure of the elongation of the spring (or compression if the term in parentheses - corresponding to stretch - is negative) along the direction of the spring and is the deformation measure of the MSS. The line of action of \( f_{\text{int}}^{s_{ij}} \) is in the direction of the spring and provides a linear relationship between the force produced for any displacement. A larger stiffness \( k \) corresponds to a stiffer spring. Since the springs can only extend in a straight line, the resolution of the mesh used in a MSS model dictates the behaviour that the body will exhibit. The internal forces acting on any one node is a sum of the spring forces of the springs that attach to that node; the springs constrain the motion of that node:

\[ f_{\text{int}}(x_i) = \sum_{j=1}^{m} f_{\text{int}}^{s_{ij}}(x_i). \]

The internal force at node \( i \) is the sum of all spring forces from springs \( j \in 1, ..., m \) that connect node \( i \) to other nodes. A consequence of this is that \( f_{\text{int}} = f_{\text{int}}(u) \): the internal forces of the entire system are dependent on the current displacement.
values of the nodes (only) and the spring stiffness coefficients $k_j$ are constant. The stiffness matrix of MSS models is much more sparsely connected compared to FEM, in direct correspondence to the connectivity information of the nodes. There are fewer connections between nodes due to springs in MSS models compared to nodes connected in elements in FEM models. The stiffness matrix assembly for MSS models is given in Appendix A.2.

### 2.5.6 Position-based dynamics (PBD)

PBD was introduced as being conceptually similar to MSS in that the body is formed of discrete point masses at nodes and a network of (linear stretch, or spring) constraints is placed upon the system. However, many other constraints have been introduced that separate PBD as a PBDM from MSS entirely. It is popular because of its robustness (the simulation can never become numerically unstable) and fast computation [35]. PBD consists of two steps: a forward update step followed by a constraint projection - or correction - step. There is no explicit internal force term as appears in (2.5); constraints instead express the inter-connectivity of the nodes. In this way, the method is position-based (rather than force-based).

In terms of the algorithm, the forward update step takes the inertial and body force terms of each node and explicitly integrates forwards in time to determine new nodal velocities. These velocities are in turn explicitly integrated to calculate the new displacement values. In the correction step, the resulting displacements are constrained by the positional constraints applied to the system. Applying the constraints is known as the constraint projection step. The constraint projection step can be multiple applications of all of the constraints at every time step. Thus, the constraint projection can comprise multiple solver iterations. However, it has
been established that the number solver iterations affects the behaviour of the material [35].

As mentioned, the constraints operate directly on the point masses in a (linearised) correction step after the forward update step. Due to the lack of internal force term, the method is not entirely physics-based. Nevertheless, the constraints have a stiffness value \( k \in [0, 1] \) that is adjustable by the user. Example constraints include *distance constraints* (inspired by the springs in MSS) that partially limits how far two point masses might separate and *volume constraints* that conserve the initial volume of each discretised element of the system. Many more constraints have been proposed [35].

The behaviour of a physical entity simulated by PBD is dependent on the time step size and the number of iterations of the constraint solver, used to constrain the displacement of the nodes. In addition to adjusting the stiffness values of the constraints \( k \), adjusting the time step and solver iteration count in turn adjust the stiffness of the system [17]. Thus, in the limit of infinite constraint iterations, the constraints become infinitely stiff [32]. Moreover, PBD does not converge to a certain solution upon mesh refinement [17]. The number of iteration of the constraint solver is treated as a material property in Section 5.4 when optimising the material properties of PBD to match the deformation of a reference NLFEM simulation.

**XPBD**

An extension to PBD, called extended position-based dynamics (XPBD) that addresses the issue of the stiffness of PBD models being dependent on both the time step and the number of iterations of the constraint solver [48]. This is achieved through making the constraints *compliant*, whereby a compliance (or inverse stiffness) \( \alpha = \frac{1}{k} \) is assigned to each constraint. In the proceeding iterations
of the constraint solver, the value $\tilde{\alpha} = \frac{\alpha}{h^2}$ is used, which shows the dependence on the time step, $h$, to constrain the motion of the nodes. The amount that each constraint constrains the motions of the nodes is constructed to depend on how much that constraint has already constrained the motion in preceding iterations [48]. Setting $\alpha = 0$ gives the original PBD formulation, corresponding to zero compliance (or infinite stiffness). The compliance is grossly related to engineering stiffness (i.e. Young’s modulus $E$) by $\alpha = \frac{1}{E}$ under small strains [48].

2.5.7 Non-physically based models

Some deformable models exist that are not physics-based but act to displace the nodes of the discretised body nonetheless, for instance shape matching [27]. The main advantage of non-physics based models comes in the computational speed of the algorithms, which do not require careful numerical integration of forces because they are not physically based. In addition do not require connectivity information (e.g. mesh edges, mesh elements, etc.) of the nodes to be stored [27]. More recent advancements have been made in data-driven methods that use deep neural networks from machine learning that combine the fast approximations of convolutional neural networks with the precision of standard solvers to give (visually) realistic results [49]. The trade-off of having high performance in these approaches comes in their lower levels of accuracy. We can quantify the accuracy using the framework presented in chapter 3 because we only require the nodal positional data.

Summary

The examples PBDMs presented so far in this Section have been physically based to varying degrees. They all aim to solve the discretised equations of motion (2.5)
on a computer and achieve this through different levels of connectivity between the nodes inside the body. More connectivity between the nodes, such as with elements in FEM compared to springs with MSS, corresponds to a more physically correct model but this is obtained at the cost of number of computations. The goal of alternative PBDMs such as corotational FEM, MSS, PBD and XPBD is to save computation time whilst still achieving similar deformations to that of the more computationally expensive NLFEM model. Chapter 3 provides a means to measure this similarity.

All of the physics-based PBDMs introduced in this Section require a forward integration step in time that takes the discretised body from its current deformed state to its next deformed state. There exist many numerical schemes to achieve this, each of varying levels of accuracy, stability and performance that directly affect the simulation of the deformation in an IVE. Some of the most common schemes are introduced in the next Section.

2.6 Numerical integration

The resulting nodal displacements in response to external loads in governed by equation (2.5). The solution of this equation requires (forward) time integration to advance the simulation into the future. In this Section we introduce numerical integration schemes of order accuracy one and of order accuracy two. Schemes of less order accuracy are less accurate but require fewer numerical computations to obtain the numerical solution. We refer the reader to the well-documented literature on the subject of numerical integration, for instance [9].

For dynamic problems as presented in (2.5), we have

$$M \frac{d^2 u(t)}{dt^2} + f_{\text{int}}(u(t), t) = f_{\text{ext}}(x, t).$$
We can write this second order ODE as two first order ODEs that can be solved one after the other:

\[
M \frac{dv(t)}{dt} + f_{\text{int}}(u(t), t) = f_{\text{ext}}(x, t)
\]

\[
\frac{du(t)}{dt} = v(t).
\] (2.27)

The derivatives can be numerically approximated using finite differences.

### 2.6.1 Finite differences

Finite differences in this context are a difference quotient used to numerically approximate a continuous derivative. There are different finite difference methods for achieving this approximation, each of which balance computational efficiency and accuracy. The most common finite difference approximation to derivatives are given in Table 2.1. We give further information, including finite difference derivations, in Section B.1 of Appendix B.

<table>
<thead>
<tr>
<th>Category</th>
<th>Type</th>
<th>Continuous analogue</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>First difference</td>
<td>Forward difference</td>
<td>(u'(t))</td>
<td>(\frac{u(t+h) - u(t)}{h})</td>
</tr>
<tr>
<td>First difference</td>
<td>Backward difference</td>
<td>(u'(t))</td>
<td>(\frac{u(t) - u(t-h)}{h})</td>
</tr>
<tr>
<td>First difference</td>
<td>Central difference</td>
<td>(u'(t))</td>
<td>(\frac{u(t+h) - u(t-h)}{2h})</td>
</tr>
<tr>
<td>Second difference</td>
<td>Forward difference</td>
<td>(u''(t))</td>
<td>(\frac{u(t+2h) - 2u(t+h) + u(t)}{h^2})</td>
</tr>
<tr>
<td>Second difference</td>
<td>Backward difference</td>
<td>(u''(t))</td>
<td>(\frac{u(t) - 2u(t-h) + u(t-2h)}{h^2})</td>
</tr>
<tr>
<td>Second difference</td>
<td>Central difference</td>
<td>(u''(t))</td>
<td>(\frac{u(t+h) - 2u(t) + u(t-h)}{h^2})</td>
</tr>
</tbody>
</table>

Table 2.1: Common finite difference approximations to continuous derivatives are presented for the scalar-valued, time-varying function \(u(t)\). These are discussed further in Appendix B.
2.6. NUMERICAL INTEGRATION

2.6.2 Integration methods

We can use the finite difference approximations presented in Table 2.1 in the time derivative quantities in equation (2.27). In particular, we use the first order forward difference equation for

\[
\frac{dv(t)}{dt} \frac{du(t)}{dt}
\]

to obtain the first order forward difference numerical approximation to (2.27):

\[
M \frac{v_{t+h} - v_t}{h} + f_{\text{int}}(u(t), t) = f_{\text{ext}}; \quad (2.28)
\]

\[
\frac{u_{t+h} - u_t}{h} = v_{t+h}, \quad (2.29)
\]

where \(v_t = v(t), v_{t+h} = v(t+h), \text{ etc.}\) Note that we have used the newly calculated value of \(v_{t+h}\) in equation (2.29).

We may use the numerical integration schemes presented in this Section seek to solve equation (2.28) for the (unknown) velocity field at the next time step, \(v_{t+h}\) and then solve equation (2.29) for the (unknown) displacement field. In other words, we can numerically integrate PBDMs forwards in time using the different numerical integration schemes presented in this Section. The integration methods differ in the (temporal) choice of where in time the displacements \(u\) are evaluated, for instance at the current time step \(u(t)\) or at the next time step \(u(t+h)\). The external forces are evaluated at the present time. We mentioned in Section 2.3.2 that the accuracy, computational performance and stability of a PBDM is dependent on the underlying numerical integration method used. This dependency stems from which choice of \(u\) is used. The most popular numerical integration schemes used in the CG literature are presented in Table 2.2. More detail is given in Section B.2 of Appendix B. Although not considered in this
work, adaptive time stepping is an option to retain stability and improve accuracy or performance.
<table>
<thead>
<tr>
<th>Scheme</th>
<th>Order of Accuracy</th>
<th>$f_{\text{int}}$</th>
<th>Velocity formula</th>
<th>Displacement formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explicit Euler</td>
<td>1</td>
<td>$f_{\text{int}}(u_t)$</td>
<td>$Mv_{t+h} = Mv_t + h(f_{\text{ext}} - f_{\text{int}}(u_t))$</td>
<td>$u_{t+h} =$</td>
</tr>
<tr>
<td>Explicit symplectic</td>
<td>1</td>
<td>$f_{\text{int}}(u_t)$</td>
<td>$Mv_{t+h} = Mv_t + h(f_{\text{ext}} - f_{\text{int}}(u_t))$</td>
<td>$u_{t+h} =$</td>
</tr>
<tr>
<td>Euler</td>
<td></td>
<td></td>
<td></td>
<td>$u_t + h v_{t+h}$</td>
</tr>
<tr>
<td>Implicit (backwards)</td>
<td>1</td>
<td>$f_{\text{int}}(u_{t+h})$</td>
<td>$(M + h^2K) v_{t+h} = $ $Mv_{t} + h(f_{\text{ext}} - f_{\text{int}}(u_t))$</td>
<td>$u_{t+h} =$</td>
</tr>
<tr>
<td>Euler</td>
<td></td>
<td></td>
<td></td>
<td>$u_t + h v_{t+h}$</td>
</tr>
<tr>
<td>BDF2</td>
<td>2</td>
<td>$f_{\text{int}}(u_{t+h})$</td>
<td>$(M + \frac{2}{3}h^2K) v_{t+h} = $ $M(v_{t} - \frac{1}{3}v_{t-h}) + \frac{2}{3}h(f_{\text{ext}} - f_{\text{int}}(u_t))$</td>
<td>$u_{t+h} =$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\frac{2}{3}u_t - \frac{1}{3}u_{t-h} + \frac{2}{3}h v_{t+h}$</td>
</tr>
<tr>
<td>Implicit trapezoidal</td>
<td>2</td>
<td>$\frac{1}{2}(f_{\text{int}}(u_t) + f_{\text{int}}(u_{t+h}))$</td>
<td>$(M + \frac{1}{2}h^2K) v_{t+h} = $ $Mv_{t} + h(f_{\text{ext}} - f_{\text{int}}(u_t))$</td>
<td>$u_{t+h} =$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$u_t + \frac{1}{2}h(v_t + v_{t+h})$</td>
</tr>
<tr>
<td>Implicit midpoint</td>
<td>2</td>
<td>$f_{\text{int}}(\frac{1}{2}(u_t + u_{t+h}))$</td>
<td>$(M + \frac{1}{4}h^2K) v_{t+h} = $ $M - \frac{1}{4}h^2K) v_{t} + h(f_{\text{ext}} - f_{\text{int}}(u_t))$</td>
<td>$u_{t+h} =$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$u_t + \frac{1}{2}h(v_t + v_{t+h})$</td>
</tr>
</tbody>
</table>
Table 2.2: Common numerical integration schemes are presented as applied to solve for the next nodal velocity and displacement values in equation (2.28). These are discussed further in Appendix B. The explicit schemes are conditionally stable and the time step value $h$ is severely limited. The implicit backwards Euler method (also known as the 1-step backwards difference formula, or BDF1) and BDF2 are unconditionally and A-stable and therefore we can afford to take larger time step values for $h$. The take-away point here is that to gain an order of accuracy in the numerical integration, more numerical work must be done in the form of either computing the stiffness matrix, $K$ and solving a linear system or storing more history values of nodal positions and velocities. $K$ represents the linear approximation of how $f_{\text{int}}$ changes at each discretised material point in response to changes in displacements $u$ at all discretised material points of the body. We find that BDF1 and BDF2 are most common in the real-time CG literature due to their unconditional stability, whilst the implicit trapezoidal method is common in FEM models due to its increased order of accuracy and energy conserving properties (because it is symplectic).
2.7 Summary

In this Chapter we have presented the PBDMs that are most popularly used to deform continuous bodies in the computer graphics literature. We remarked in Figure 2.2 that the PBDMs all follow the same general solution procedure that require numerical integration forwards in time but differ in how they model both the measure of deformation and the internal forces.

Our presentation of PBDMs starts with finite element methods used to numerically approximate the solutions to the equations of continuum mechanics. In the NLFEM model we measure the deformation using nonlinear displacement gradient terms in the strain measure and model the stress-strain relationship nonlinearly to give different material responses under different changes in external loads. This is a very computationally intensive procedure and is generally not suitable for real-time CG applications. In the linear approximation to NLFEM we linearly approximate the strain measure (truncating the nonlinear displacement gradient terms) and use a linear stress-strain constitutive law. This saves a lot of computation time per update step, making it suitable for real-time CG applications. However, it can produce visual artifacts in the form of volumetric growth of the deformed body when large stretches and large relative rotations occur. Corotational FEM models are a suitable compromise between NLFEM and LFEM in that they remove the visual artifacts due to large relative rotations but are more computationally efficient than NLFEM. These three FEM models were compared to provide a visual summary in Section 2.5.4.

The next PBDM presented was MSS models that model the particle interactions of a discretised continuous body much more simply. The internal particle interactions are modelled using springs that are linear stretch constraints that obey Hooke’s law. This simplification of the particle interactions still provides
visually plausible simulations and since the computations are simple, this PBDM is suitable for real-time CG. We quantify the accuracy of MSS in Chapter 3. In addition to MSS models, we presented PBD models that do not model internal forces but instead impose constraints on what the displacements of the nodes can be. In contrast to other PBDMs, PBD models do not model internal particle interactions using an elastic potential energy function but instead use positional constraints that act directly on the calculated nodal displacements at each update step. The upshot of this is that PBD models are unconditionally stable at the expense of not being derived from physical balance laws, which makes the models less accurate in general. The accuracy of PBD models is quantified in Chapter 5 for 3D cantilever and cloth scenarios.

Finally, we presented common numerical integration schemes used in the CG literature. Should a PBDM employ a certain numerical integration scheme, the performance, stability and accuracy of the deformation directly corresponds to the scheme used. Numerical precision errors, although not considered in this work, can further leave to more errors in the accuracy and affect the stability of the simulation also.
Chapter 3

Quantitative Validation of Deformation Histories

In this Chapter we present the quantitative validation of deformation histories (QVDH) procedure that quantifies the agreement of a deformation with a reference deformation over time. The produced quantity is in the range \([0, 1]\) and is a measure of accuracy of the deformation history being evaluated. In addition, being able to obtain this quantity allows us to directly compare two different test deformation histories against each other in terms of accuracy. The work in this Chapter directly builds upon validation techniques that we have observed in the CG literature.

In Chapter 1 we presented a component-based software framework that forms the basis for validation of PBDMs in CG as found in the real-time CG literature. The framework facilitates the evaluation of accuracy of a deformation of a PBDM based on a single chosen end-state. The component diagram for this is illustrated in Figure 1.6. We discuss these literature pieces in Sections 3.1 and 3.1.2, before discussing how time-evolving quantities are measured in other engineering disciplines, namely vehicle simulation and vehicle safety in Section 3.1.3. In vehicle safety, time-evolving quantities are stored together to produce two discrete time
signals. These signals are then compared to quantify how similar they are. This is a *temporal* quantification and in this Chapter we adapt it to the validation of deformation histories in real-time CG. We do this to provide an error metric that quantifies a(ny) deformation history of any PBDM against a reference deformation history. We use NLFEM as the reference PBDM as is traditionally done in engineering practice, making sure to use a mesh that is sufficiently finely discretised to minimise errors in the solution of the NLFEM model due to discretisation. We carry out the QVDH procedure in a component-based software framework, which builds on that of Figure 1.6 in Chapter 1 and we present the framework in Section 3.2. The interfaces of the components of the software framework are presented in the next Chapter.

As example applications of the software framework, we evaluate two popular PBDMs from the real-time CG literature in a cantilever beam scenario in Section 3.3. Both PBDMs are mass-spring systems (MSS), which was introduced in Section 2.5.5. The first MSS solves the equations of motion using an explicit symplectic Euler scheme (see Section 2.6) that does not dissipate energy but necessitates a very small time step in order to stay stable. The second MSS poses the equations of motion as an energy minimisation problem. This means that we expect the computational performance of the solution procedure to be adjustable by performing fewer (or more) iterations of the solver at an expense (or gain) of accuracy. We present the results in Section 3.3 and obtain ratings of accuracy for each. We find that the ratings of accuracy are in agreement with the expectations we have about the second MSS. Further, we can directly compare the accuracy of each MSS against one another. Limitations of the approach are highlighted in Section 3.5 in which we find that performance of the QVDH procedure is bound by the performance of the reference PBDM, which is simulated in real-time. Thus the QVDH procedure is not real-time either. It also requires a lot of data to be
stored in addition to that of each PBDM used in the framework, such as in the discrete time signals. Finally, the effectiveness of the accuracy evaluation depends on the choice of displacement measure used. We find that using a single scalar value at each time step to represent how much an entire discretised body has displaced can hide particular detail if not chosen carefully.

3.1 Related work

We find that it is common for validation of PBDMs in CG to be only qualitative, which is something that has been expressed in surveys of the real-time CG literature by Nealen et al. [2] and Muguercia et al. [43]. Often the acceptance criteria for accuracy is through visual plausibility [35][2]. A disadvantage of qualitative validation is that it is subjective; the validity of a PBDM is dependent on the senses of the user, which are not always reliable. Bender et al. [4] note that it is difficult to detect subtle abnormalities in a simulation through senses alone and Nusseck et al. [10] remark that reliable sensory validation requires full concentration throughout. However, the qualitative validation of a deformation by a PBDM from an expert in the field (a subject matter expert, SME) [50] must also retain its significance and cannot be disregarded just because the validation obtained is qualitative. Thus, any quantitative validation ought to be consistent with the qualitative validation of any such SME [51]. Nevertheless, [1][43] acknowledge that there is room for more concrete assessment and, furthermore, [43][52] acknowledge the need to validate results with data from the real world via a standard set of scenarios. As we mentioned in Section 2.3.3, the NLFEM applied to acquire numerical solutions across a suitably high resolution mesh serves as the gold standard for accuracy amongst PBDMs. This means that it can be taken as being true to the real-world and any quantitative comparisons that we make against
this gold standard PBDM provide a means of validating the simulation as being accurate or not.

We find that quantitative validation techniques have seldom been used in interactive CG. In evaluating the material properties in a PBDM, both the Hausdorff distance [53] and mean absolute error (MAE) of displacements of points on the body are used to validate the results of stretching a cloth. Bender et al. [17] compare simulation data of a NLFEM model against real-world test data. These measures are taken at the end of the stretching (only) and so, being a single time instance, do not serve as a temporal evaluation of accuracy. The deformation history is not considered in the measurement of accuracy. To compare the time-evolution of such accuracy measurements, one would evaluate the dynamic, temporal agreement of a deformation history of a PBDM against a reference deformation history. One can imagine a scenario whereby the end deformations are in good agreement even though the deformation histories are not, such as in the trivial example of Figure 3.1. A field in engineering that assesses the temporal agreement of reference and test data is vehicle safety, which we discuss in Section 3.1.3.

### 3.1.1 Quantitative validation

The establishment of quantitative validation requires the evaluation of some quantities to show agreement. Sarin et al. define an error measure as a quantity associated with differences of a particular feature of a time series, whereas an error metric is an overall value of discrepancy between time series [51]. From here, an error measure will refer to a single quantification of agreement between two measures. This includes a quantification of agreement at a single time instance that alone is not suitable for quantitative validation. An error metric will reflect
3.1. RELATED WORK

Figure 3.1: In this trivial example, the deformations produced by the reference and test PBDMs have matching end-states. However, their deformation histories do not match. The deformation by a test PBDM would be incorrectly validated as being accurate if only the state at $t = T$ was considered in the evaluation of accuracy.

<table>
<thead>
<tr>
<th></th>
<th>$t = 0$</th>
<th>$t = T/2$</th>
<th>$t = T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>reference</td>
<td><img src="image1.png" alt="image" /></td>
<td><img src="image2.png" alt="image" /></td>
<td><img src="image3.png" alt="image" /></td>
</tr>
<tr>
<td>test</td>
<td><img src="image4.png" alt="image" /></td>
<td><img src="image5.png" alt="image" /></td>
<td><img src="image6.png" alt="image" /></td>
</tr>
<tr>
<td>Validated?</td>
<td>✗</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

The agreement of the entire (deformation) histories; an error metric is a temporal evaluation of accuracy. Any kind of model validation has unavoidable uncertainties that arise either due to chance or they are because we do not have the knowledge to exhaustively and precisely model a scenario. As an example, a physically based deformable model that is the gold standard for accuracy in engineering practice (such as NLFEM) cannot be validated for accuracy in scenarios for which real-life data does not exist. Instead, the model is assumed to be accurate even though we cannot be certain. To acknowledge these uncertainties gives a stochastic validation metric whilst ignoring them gives a deterministic validation metric [51][54]. Stochastic validation metrics allow the evaluation of computational models in both tested and untested design regions due to their necessitation of comparing probability distributions [54][55][56]. Moreover, Sankararaman et al. [57] provide a means of evaluating how uncertainty propagates around a full, complex system. They illustrate how each subsystem or component of a system contributes to the final uncertainty quantification, which is evaluated through
the integration of said components into a (Bayesian) network. They say that the components interact by one of two ways. A Type-I interaction between two components is when the output of one component (whose produced output data is classified as uncertain) becomes the input data to a higher level component. A Type-II interaction is when another model acts as a surrogate and replaces a higher fidelity model. A surrogate model here is a computationally inexpensive model in comparison to the original high fidelity one, with the hopes that this model can achieve the goals of the original model. This is the process undertaken in researching and developing new PBDMs in real-time CG: the tested PBDM is a surrogate PBDM and we hope that its output data is in agreement with the data produced by a (more computationally expensive) gold standard PBDM. The process of directly, quantitatively comparing data (through the use of error measurements) is a deterministic procedure because both sets of data are already available, even though those data sets may have been obtained stochastically. Deterministic validation metrics are considered in this thesis.

Measured errors in numerical computation can arise due to both model and discretisation error [58], as well as due to solver error in numerically solving linear systems of equations. Model error is the error induced by the mathematical formulation of a physical problem. Such errors that arise include the errors from the approximations made in the chosen PBDM and numerical integration scheme, as we presented in Chapter 2. Discretisation error is the error arising due to the discretisation (approximation) of a mesh that is modelling a continuous volume. An increase in resolution might lead to the convergence of a numerical solution; this is the minimisation of discretisation error [58]. Error estimates work to minimise the discretisation error in finite element solutions [42]. Once minimised, any other error against some ground truth is primarily due to modelling error. Validation metrics that quantify discretisation error are deterministic because
3.1. RELATED WORK

the experimental data sets are being compared with certainty (even though the data sets may have been obtained stochastically) [57], so the discretisation error is minimised deterministically. The error that is quantifiable in the presented software framework arises from both model and discretisation errors; it is not a tool for diagnosing sources of errors.

3.1.2 Validation in surgical simulations

We find that the methods of deformation validation in surgical simulation are deterministic, quantitative and performed at a single point in time, for instance in works such as [18][14][15][19][59][22][23][60][61]. These works compare computational simulation data against real-life test data and then verified based on a chosen end state, for example when the simulation comes to rest. The error measure operates on displacements of points between data sets and is typically a central tendency (e.g. the mean value) coupled with the variation [18][14][15][19], maximum error [15][19] or some variation of the Hausdorff distance [59][22][23]. This overwhelming focus on end-state validation ignores the histories of deformations. Nevertheless, test data exists in the form of the so-called Truth Cube for real-time soft tissue deformation [38][28]. The test data is available as nodal positions for a series of quasi-static loads. In a similar fashion to surgical simulation, we find that literature concerning robotics follows the same patterns. We leave relevant literature pieces (such as [62]) to Chapter 5.

We find that works in the literature rarely record time-evolving error measures. One example instance we found was in the work of Courtecuisse et al. [19], who evaluate the trade-off between accuracy and performance of a PBDM by plotting the root-mean-square error (RMSE) (of nodal displacements of the test PBDM against a reference PBDM) against time. Whilst this allows for visual, qualitative
confirmation, there is no translation of this error measure history into a quantity that might be used to compare two time series objectively, as will be done here.

Stochastic validation measures are seldom used. Deformable image registration is used to match images of tissue before and after the advancement of disease [24]. The LFEM with adaptive mesh refinement is used to find the best deformation of the (mesh corresponding to the) before image that gives rise to the after image. The matching is done using three “popular similarity metrics”: MSE, normalised correlation (NC) and normalised mutual information (NMI). These measures are descriptive statistics and allow us to evaluate the question “is there any difference?”. The use of inferential statistics allow us to evaluate the question “is the difference meaningful (beyond the sample size)?” A t-test is used that gives the ratio of variance between groups against the variance within groups; it is a signal to noise ratio based on the finite sample size when we do not know the distribution of the entire set. The hope is to minimise the likelihood of the results being due to chance; the sample size is hopefully reflective of the entire data set. Of course, there will always be some amount of certainty. Classical hypothesis testing is used to dismiss the null hypothesis that the MAE of distances to the reference image of the proposed method are worse than existing methods. For each of the three similarity metrics, the $p$-value is “$p < 0.05$”. It is worth noting, however, that unless this value is closer to 0 than 0.05, the results should only be interpreted as mildly statistically significant. Nevertheless, the results looks promising and the uncertainty of data acquisition is addressed.

An entire deformation history requires performing an error measurement at each time step. A step needs to be taken to transform a pair of collections of error measures over time into an error metric. As mentioned at the start of Section 3.1, any sort of discrepancy between an entire pair of time series that gives rise to some kind of quantification metric should also reflect the views of a SME (subject
matter expert). As an example use of validation by SMEs, a survey is conducted by Sui et al. in [21]. They model a deformation using PBD and have its output evaluated by surgeons, who give 90% mean satisfaction with a 5 – 10% standard deviation. A high mean score with a comparatively small spread, sourced from SMEs, provides very good validation that lies somewhere between qualitative and quantitative. The ratings from SMEs are the most informed and reliable, and should be the most objective. As such, error metrics that are developed to reflect the opinions of SMEs should be used. This is precisely what is done in the engineering field of vehicle safety.

### 3.1.3 Validation in vehicle safety

Barbat et al. [50] state that qualitative validation should be based on the knowledge of subject matter experts (SMEs). A problem with this is that each design iteration - be it large or small - requires the constant cooperation of developers and SMEs. This can be a slow back and forth conversation that can hinder productivity should the SME not be readily available.

In some literature works on vehicle safety, for instance [51] and [63], some signals (plotted as curves on a graph) are quantitatively compared and rated to show the level of agreement between them. The purpose of this rating is to compare two *unambiguous signals* that are not directly related. This means that some measure is plotted against time, once for (real-life) test data serving as the reference and once for computational data serving as the comparison, resulting in two unique discrete time signals. The signals are compared and the level of agreement between them is calculated. Crucially, the calculated score reflects the opinions of SMEs and, since the field is vehicle safety, the selection of parameters in the calculation of signal agreement is strict so as to comply with the obvious
safety constraints that such a field demands.

We turn our attention to metrics that quantify the agreement between pairs of signals. The CORelation and Analysis (CORA) metric of Gehre et al. [64] and the Enhanced Error Assessment of Response Time Histories (EEARTH) metric of Barbat et al. [50] are two standardised, leading metrics used in vehicle safety evaluations. CORA uses two independent sub-ratings. The first is a corridor rating that assesses, at each sample point of the signal, how close the corresponding point on the comparison curve is through the use of “corridors” around the reference curve. The second is a cross correlation rating that directly assess the characteristics of the curves such as magnitudes, shapes and phase shifts. The idea is that each sub-rating makes up for the shortcomings of the other. EEARTH enhances another error metric of Sarin et al. [51] called EARTH that uses three sub-ratings relating to phase, magnitude and curve topology. Parameters for both CORA and EARTH are tuned by the user for the application. EEARTH has its parameters either fixed by SMEs or automatically calculated depending on the reference curve and produces a rating in the range $[0, 1]$, with 1 being the best score for matching signals.

The ISO18571 develops an error metric that utilises the best features from CORA and EEARTH, which are the leading metrics in the field [50]. The magnitude ($e_M$), phase ($e_P$) and topological ($e_T$) error ratings of EEARTH and the corridor rating ($e_C$) of CORA form a weighted linear combination to give:

$$e_{ISO18571} = 0.4e_C + 0.2e_M + 0.2e_P + 0.2e_T \in [0, 1].$$

(3.1)

The different components provide measured as follows. More details may be found in [50].

- $e_C \in [0, 1]$ measures how physically close the discrete points of the DTS are
3.1. RELATED WORK

to the other DTS. This is done through the use of two “corridors” around the reference signal. The score at each point of the test DTS is scored as shown in Figure 3.2. The average of all of these scores then gives $e_C$.

- $e_M \in [0, 1]$ measures the similarity of the maximum magnitude (i.e. the amplitude) of each signal.

- $e_P \in [0, 1]$ measures how in-phase the two signals are. This is achieved through dynamic time warping (DTW) that aims to fit the test signal to the reference signal in the case that they are out of phase or at differing frequencies.

- $e_T \in [0, 1]$ measures the slope error by forming a (maximum of) 10 piecewise linear spline approximation to the signals between the DTS and comparing the gradients of these two piecewise linear splines.

The linear weights in (3.1) are chosen to be complementary to one another. The corridor rating is given the largest weighting as it measures the physical closeness, although the score that it produces is very sensitive to phase errors. For this reason, the DTW procedure of $e_P$ is applied first before all of these sub-ratings are calculated so as to not obfuscate the potential scores of the other sub-ratings due to phase differences. This sensitivity is one of the biggest drawbacks of the metric. The implementation details are not entirely clear either, however a black box is instead made available by the authors. Moreover, it is not entirely clear why, where or how the ratings score badly. In this way, the metric is not a diagnostic tool, but rather serves to identify that some error exists by considering many different characteristics of the two signals. Nevertheless, the following thresholds are set by SMEs and remain valid only when the parameters of the individual error measures are unchanged from the ISO standard, which are set by the SMEs
Figure 3.2: The CORA metric measured how close the test signal is to the reference signal through the use of two corridors. The final result is the average of all DTS sample points.

[50][65][66]:

<table>
<thead>
<tr>
<th>Rating, $R$</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0.94 &lt; R \leq 1.00$</td>
<td>Excellent</td>
</tr>
<tr>
<td>$0.80 &lt; R \leq 0.94$</td>
<td>Good</td>
</tr>
<tr>
<td>$0.58 &lt; R \leq 0.80$</td>
<td>Fair</td>
</tr>
<tr>
<td>$0.00 \leq R \leq 0.58$</td>
<td>Poor</td>
</tr>
</tbody>
</table>

An example is illustrated in Figure 3.3.
3.2 QVDH SOFTWARE FRAMEWORK

The base component-based software framework presented in Chapter 1 facilitates the quantitative validation of a pair of body deformations at a single instance in time. This is not a temporal evaluation of accuracy. To perform a temporal, quan-
We extend the base software framework outlined in Figure 1.6 to allow for accuracy quantification over time rather than at a single time instance.
quantitative validation of accuracy, we require a component to perform a displacement measure - which will be used to measure the displacement of each body at every update step - and an error metric component that quantifies the agreement of two series of time-evolving displacement measures. Both of these components are defined in Section 3.1.1. We also need components that store the time series of displacement measures for each body. We note here that, if compared PBDMs are updated at different time steps, then we (linearly) interpolate between the results to obtain values at each and every time instance of either PBDM. In practice, however, we choose the time steps to be a multiple of one another.

We implement these functional parts as components, which extends the initial framework from Chapter 1. The design of the resulting component-based software framework is presented below. In this Chapter we present the results of the using the framework and save the particular implementation details of the components for Chapter 4. To use the framework, we must first select, specify and implement all components. We acknowledge that whilst the component chosen in this thesis may not provide the best evaluation possible, a key feature of our framework is that the user can substitute components where appropriate without changing the functionality of the framework.

### 3.2.1 Displacement measure component

The displacement measure component quantifies the displacement of a PBDM at a single instance in time, relative to some reference state. The calculated quantity is a scalar value. The reference state will thus be the undeformed positions of the bodies in that particular scenario. As discussed in Section 3.1.2, variations of the Hausdorff distance are popular choices for error measures in the literature when some shape matching algorithm is used. The directed Hausdorff distance is
defined in [53] as:

$$d_{DH}(A, B) = \max_{a \in A} \min_{b \in B} ||a - b||,$$  \hspace{1cm} (3.2)

where \(A\) and \(B\) are sets of positions in Euclidean space and so \(|| \cdot \||\) is the Euclidean norm between (positions of) points \(a \in A\) and \(b \in B\). This measure is the greatest of all distances from each point \(a \in A\) to the closest point \(b \in B\) and can be thought of here as how much the test and reference bodies overlap (a measure of 0 is a perfect overlap). Due to its popularity in shape matching, the bidirectional Hausdorff (BDH) distance [53] is used in the example implementations of the framework in Section 3.3:

$$d_{BDH}(A, B) = \max(d_{DH}(A, B), d_{DH}(B, A)).$$ \hspace{1cm} (3.3)

Alternative choices are possible and require a simple substitution of the Displacement Measure component. For example the modified Hausdorff (MH) distance [67] in place of (3.2) in (3.3). We might do this if the deformation produces nodal positions that are classed as outliers but we can ignore them for whatever reason. The BDH distance is sensitive to outliers but a MH distance suppresses this sensitivity by taking an average over the nodal displacement values:

$$d_{MH}(A, B) = \frac{1}{N_a} \sum_{a \in A} \min_{b \in B} ||a - b||.$$

3.2.2 Error metric component

For each deformable body, we store the displacement measures obtained using the Displacement Measure component at every update step in a discrete time series. Each series serves as a signal and we compare two signals to produce a final rating \(\epsilon_{ISO18571}\) as introduced in Section 3.1.3. This rating is for the tested scenario and
serves as the measure of accuracy for the tested PBDM within that scenario. We reiterate that a key feature of the framework is that other components can easily be used instead. For instance, an error metric that calculates the difference in areas under the signals when the signals are plotted as curves on a graph.

### 3.2.3 Scenario component

We observed the following common features of the modelled scenarios in literature of CG.

1. **Test data**: does either an (exact) analytical solution or experimental data exist against which we can compare our simulated results?

2. **Complexity**: are interesting deformations covered? Are the scenarios sufficiently complex enough (i.e. nonlinear deformations) to reflect real-life deformations?

3. **Geometric simplicity**: are the continua easy to model in a PBDM? Can a mesh be easily constructed?

4. **Duration**: over what time duration should the simulation remain accurate?

We therefore choose scenarios that satisfy all of these requirements. Example scenarios in the literature were given in Table 1.1. We observed that popular choices are cloth and cantilever beam scenarios, with external loads coming from gravity and user interaction. User interaction is typically collisions due to the user controlling an additional entity within the virtual environment. We observed that organ deformation was common in surgical simulation, which are subject to a lot of user interaction. The factors listed above are all addressed for these scenarios, although the duration of the scenarios is not always specified because
the reviewed literature pieces focused their attention to the final state at which their PBDM was evaluated for accuracy.

### 3.2.4 Deformable model components

We must provide two PBDMs. The first is a test PBDM that is under evaluation within the framework. We provide an implemented test Deformable Model component in the framework that produces a new (possibly deformed) physics state at each update step that will be evaluated (at every update step). The reference data in the literature is often from real-world data or it can even be an analytic solution, should one exist in the particular scenario. In scenarios for which no analytic or test data exists, we use a suitably accurate PBDM as the reference PBDM component instead that provides reference data at every update step. We use the NLFEM (using a St. Venant-Kirchhoff strain energy function) as the reference PBDM in the evaluations made in the software framework in Section 3.3.2. It is often used in the literature as a reference PBDM for validation. Its suitability as being a candidate reference PBDM is demonstrated in Section 3.3.1. We note that a PBDM could qualify as being a candidate reference PBDM in that scenario if it achieves an ISO18571 error metric rating that grades as “excellent” (see Section 3.1.3) when compared against ground truth reference data.

### 3.3 Results

We use the software framework to quantitatively evaluate the accuracy of two MSS models (popular in the CG literature) in Section 3.3.2. In Section 3.3.1, we show that the NLFEM model is a suitable reference PBDM and we outline a particular test scenario that will be used in the evaluations of Section 3.3.2. We give analysis of the results of the evaluations in Section 3.4.
3.3. RESULTS

3.3.1 Test scenarios

As mentioned in Section 3.2.4, if we have neither an analytic solution nor a set of real-life test data exist for a scenario, we use a suitably accurate PBDM as the reference PBDM to provide the reference data. Thus, we evaluate the NLFEM in the framework in a steady state 2D cantilever beam scenario for which an approximate analytic solution can be derived. We do this for different levels of discretisation of the 2D mesh in order to minimise discretisation error. Ultimately, our goal is to demonstrate the high accuracy of the NLFEM and to justify its use as a suitable reference model. Following this, we present an unsteady 3D cantilever beam scenario that adheres to the scenario requirements of Section 3.2.3, together with the NLFEM configuration that will serve as the reference PBDM. We use this scenario and reference PBDM combination in evaluating test PBDMs in Section 3.3.2.

Establishing a gold standard: Steady 2D Cantilever Beam

The classical solid mechanics problem of the bending of a 2D cantilever beam subject to a constant pressure loading, $P$, on its upper face is as in Figure 3.5. The balance of linear momentum is given by (2.15) as

$$\rho_0 \frac{\partial^2 \mathbf{u}}{\partial t^2} = \nabla \cdot \mathbf{\sigma} + \mathbf{f}_{\text{ext}},$$

where $\rho_0$ is the density of the undeformed continuum, $\mathbf{f}_{\text{ext}}$ are the external loads, $\mathbf{u}$ is the displacement and $\mathbf{\sigma}$ is the second Piola-Kirchhoff stress tensor. An approximate steady state solution for the stress field can be constructed from the following Airy stress function. The steady state means that there are no external loads and no inertial forces: $\frac{\partial^2 \mathbf{u}}{\partial t^2} = \mathbf{f}_{\text{ext}} = 0$. The Airy stress function is valid for
small pressures in the steady state [42]:

\[
\Phi(x_1, x_2) = ax_1^2 + bx_1^2x_2 + c(x_1^2x_2^3 - \frac{1}{5}x_2^5) + dx_2^3,
\]

(3.5)

where

\[
a = -\frac{1}{4}P, b = -\frac{3}{8} \frac{P}{H}, c = \frac{1}{8} \frac{P}{H^3}, d = \frac{1}{20} \frac{P}{H}.
\]
The displacement field can be recovered as ($\mu$ is a Lamé constant, derivable from $E$ and $\nu$):

$$u_1 = \frac{1}{2\mu} \left( (1 - \nu)(2ax_1 + 2bx_2(x_1 - L)) + 2cy(x_1^3 - x_1x_2^2 - L^3) + 6dx_2(x_1 - L) \right)$$

$$+ 2a(\nu L - x_1) - 2bx_2(x_1 + L) - 2cx_1x_2^3 \right);$$

$$u_2 = \frac{1}{2\mu} \left( (1 - \nu)(2ax_2 + b(x_2^2 - x_1^2 + 2Lx_1 - L^2)) + c(3x_1^2x_2 - \frac{1}{2}(x_1^4 + x_1^4 + 3L^4) + 2L^3x_1) + 3d(x_2^2 - x_1^2 + 2Lx_1 - L^2)) \right)$$

$$+ b(3L^2 - x_1^2 - 2Lx_1) + c(x_1^4 - 3x_1^2x_2^2) + 3d(2Lx_1 - x_1^2 - x_2^2 - L^2))\right).$$

The pressure is incremented at each timestep to give a series of quasi-static loads. The length is set to $L = 5$ and the height to $2H = 1$ so as to nondimensionalise all lengths. A NLFEM solution is computed using oomph-lib [42], with $E = 250\text{Nm}^{-2}, \nu = \frac{1}{3}$. To use these model parameters in the St. Venant-Kirchhoff strain energy function, they are converted to the Lamé constants via

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \mu = \frac{E}{2(1+\nu)}.$$  

The material is very soft, which gives a large deformation under the applied pressure load. This is compared against the analytic solution, serving as the reference. To eliminate discretisation errors in the NLFEM model, the ISO18571 error metric rating of Section 3.2.2 is calculated for different discretisations of the body of the beam. Figure 3.6 shows this. We create a time-evolving simulation by increasing the pressure load on the upper surface at each time step to simulate a (quasi-)moving beam scenario. This allows us to plot two signals over time based that displays the evolution of the measured Hausdorff distance of the beam from its rest (pressure-free) state. We then measure the
similarity of the curves using ISO18571.

Figure 3.6: The 2D cantilever is discretised using rectangular elements. We adjust the number of nodes in the $x_1$ and $x_2$ directions, which changes the discretisation level of the mesh. We plot a graph showing the ISO18571 rating, $q$, against the independent variables $x_1$ and $x_2$. The point circled in red is the minimum discretisation for a score of at least 0.97. At higher levels of discretisation, the value increase of the rating $q$ is negligible, implying that discretisation error has been minimised. We take the solution at the highlighted level of discretisation as being as accurate as the real-world.

The results show that for discretisations of at least $x_1 \geq 15, x_2 \geq 3$ nodes, the result will be at least $q = 0.97$. A score of 0.97 is excellent in accordance with the ISO18571 grade boundaries. For even finer meshes, the score is at least 0.97, showing that result has converged and discretisation error has been minimised. Whilst mesh refinement procedures using a posteriori error estimates can (and will) be used in scenarios for reference model calculations, here the sensitivity
of the framework to a purposeful increase in mesh resolution can be observed through the corresponding error metric scores in Figure 3.6. The high scoring error metric scores validate the NLFEM model to be used as a reference PBDM in future evaluations in the framework.

Unsteady 3D Cantilever Beam

We first test the unsteady simulation of a 3D cantilever beam. The geometry is the same as 3.3.1 but is extended to three dimensions, such that the \( x_3 \) axis points up in this Cartesian basis. Since the problem is unsteady, the inertia term \((\rho_0 \frac{\partial^2 u}{\partial t^2})\) is now included, whilst the pressure loading from the 2D scenario is removed. Instead, external loading is in the form of a body force of \((0, 0, -1)^T\). This has the effect of the beam being previously steady in its initial position and instantaneously being subjected to the body force at the start of the simulation. The body is perfectly elastic; no damping occurs and the body oscillates indefinitely. The approximate analytic solution, \((3.6)\), is for steady state only and is therefore no longer valid. The undeformed body is a parallelepiped shape and simple to recreate. We run the simulation for 5 seconds to allow for a significant deformation and see that the beam oscillates. See Figure 3.7.
CHAPTER 3. QVDH

Figure 3.7: The modelled 3D cantilever is fixed at one end and deforms under gravity. The beam oscillates indefinitely because no energy dissipation (velocity damping) is modelled. Top left: the beam at rest. Top right: the beam after 1.75 seconds. Bottom left: the beam after 2.5 seconds, which corresponds to the most displaced position of the simulation. Bottom right: the beam after 5 seconds, which corresponds to the beam temporarily returning to its rest state before oscillating again.

The cantilever is $4 \times 1 \times 1 \text{m}^3$ and the material is both homogeneous and isotropic, with elastic modulus, $E = 400 \text{Nm}^{-2}$ and Poisson’s ratio, $\nu = \frac{1}{4}$. Thus the material is very soft. At fixed time steps, $h$, we advance the simulation by $h$. We calculate the bidirectional Hausdorff distance of the beam to its initial position at each time step. We discretise the reference NLFEM, serving as the gold standard, with
the use of the $Z^2$ error estimator [41][42] with strict error thresholds (minimum and maximum permitted errors of order $10^{-4}$ and $10^{-3}$, respectively [42]) to give a non-uniformly discretised mesh of greater than 4000 nodal points. In Figure 3.8 we compare solutions from different, unrefined and uniform resolutions with this gold standard. We can observe that as we more finely discretise the mesh, the error rating converges to $e_{\text{ISO18571}} = 0.98$. Therefore, in this example, an increase in error metric rating corresponds to a decrease in discretisation error.
Figure 3.8: **Above**: A graph showing the convergence of the NLFEM model when simulating the 3D cantilever beam scenario. The arrow represents an increase in mesh resolution, whilst the solid line represents the resolution with discretisation error minimised through the use of $Z^2$ error estimation. **Below**: The ISO18571 error metric scores of the signals in the above graph against the reference signal. Additional intermediate resolutions are included. The colours are for consistent mesh resolutions in each graph. We can observe that discretisation error is minimised as the mesh is refined to discretisation of $17 \times 3 \times 3$ nodes and finer because the error rating converges towards 0.995.

It remains to show what the minimum simulation updates per second can be
without sacrificing accuracy. Whilst this does not have an effect on the accuracy, if we perform fewer updates per second whilst still maintaining accuracy then we will use fewer computational resources. Figure 3.9 suggests that we can use a time step of $\frac{1}{15}$s whilst maintaining sufficiently high accuracy.

Figure 3.9: The effect of varying the time step size on the measured Hausdorff distance of the displacement of the $13 \times 3 \times 3$ mesh for the NLFEM 3D cantilever beam scenario from its undeformed state. The solid black indicates value of convergence for the trialed timesteps.

### 3.3.2 Test PBDM evaluations

We evaluate two MSS models in the software framework. The MSS models are popular PBDMs in the CG literature. As outlined in Section 3.3.1, we choose the unsteady 3D cantilever beam as the scenario and the NLFEM model as the reference PBDM.
**Evaluation 1: MSS using explicit symplectic integration scheme**

We first evaluate a MSS model (Section 2.5.5) that uses an explicit symplectic (or semi-implicit) Euler integration method (Section B.2) in the proposed framework. We discretise the mesh into cubic unit cells as in Figure 3.10.

![Figure 3.10: A 8 × 3 × 3 node parallelepiped mesh with cubes as unit cells (or elements). The springs lie across the edges and face diagonals of each unit cell.](image)

There are discrete masses at each corner of the cube with springs along each edge and face diagonal. In this setup, the (linear) springs all have the stiffness coefficient $k$ arising from the identity provided by Kot *et al.* in [68]: $E = 2.5 \frac{k}{a}$, where $a$ is the length of the cubic unit cell. It is reiterated that the modelled material is homogeneous, isotropic and has a Poisson’s ratio $\nu = \frac{1}{4}$.

Using a cubic unit cell for the beam can, if the discretisation is of low enough resolution, cause the beam to (incorrectly) not keep its neutral plane [68]. The neutral plane is the surface between the outer surfaces (that are under compression or tension) that is not under stress. We can overcome this by constructing a smaller mesh (of one fewer node in each Cartesian coordinate direction) to lie inside the outer mesh. We position the inner mesh so that each node of the smaller mesh lies in the centre of each cubic unit cell of the outer mesh. This concentrates the springs around the neutral plane of the beam. See Figure 3.11.
3.3. RESULTS

Figure 3.11: The smaller mesh (dark grey with red linear spring constraints) inside the outer mesh (lighter grey) provides more resolution per unit length. The red spheres represent fixed nodes (nodes with displacement boundary conditions equal to zero displacement).

The explicit symplectic Euler method is conditionally stable due to the CFL condition (see Section B.2) and the problem setup results in a numerically stiff system [69]. The effect of this is that the choice of time step $h$ depends on the nodal masses $m$ and the stiffness of the springs $k$:

$$h < C\sqrt{\frac{m}{k}},$$

where $C$ is a constant [27]. The insight of this inequality is that the rate at which the effects of internal particles’ interactions propagate through the system is too fast for the linear momentum to be balanced at this level of discretisation when using the MSS. Force propagation must be to neighbouring nodes only in explicit integration schemes. The choice of a smaller time step $h$ means a smaller magnitude internal force per update step. The integration is performed at fixed time intervals of $h = \frac{1}{120}$ s to accommodate this. We plot the results for different discretisations against the reference NLFEM in Figure 3.12. At finer resolution discretisations beyond $17 \times 5 \times 5$ nodes, the system is unstable because a higher
level of discretisation means the lumped nodal mass values are in turn less, which restricts the time step $h$ further. Either a change in material properties (the spring stiffness values $k$) or the adoption of a different integration method is needed if either a higher resolution discretisation of the mesh or a greater time step is desired whilst retaining numerical stability. The ratings are greatest for the discretisation of $17 \times 5 \times 5$ nodes: $q_{\text{ISO18571}} = 0.985$, which is excellent according to ISO18571.

**Evaluation 2**

Our second evaluation using the proposed framework is the same MSS as in Section 3.3.2, except now the solution procedure is based on energy minimisation. The equations of motion are solved using a so-called variational implicit Euler method [70]. The Euler-Lagrange equations of motion (2.5) in an implicit form in terms of the displacements are

$$M \frac{u_{t+h} - 2u_t + u_{t-h}}{h^2} = f_{\text{ext}} - f_{\text{int}}(u_{t+h}),$$

(3.7)

where the nodal acceleration is numerically approximated using a second order forward difference. In this way, the displacements are calculate first and then the velocity is calculated. This is slightly different to the procedure outline in Section 2.6 in which we first calculate nodal velocities and then nodal displacements, but it leads to the same solution. As per Liu et al. in [71], we set $w := 2u_t - u_{t-h}$ and rearranging equation (3.7) gives

$$M(u_{t+h} - w) = h^2(f_{\text{ext}} - f_{\text{int}}(u_{t+h})).$$

(3.8)
3.3. RESULTS

Figure 3.12: The MSS is integrated using an explicit symplectic Euler method. **Above:** The error measure is plotted against the reference NLFEM for different resolutions. **Below:** The ISO18571 is used to score the test (MSS) signals against the reference (NLFEM) signal.
The solutions of equation (3.8) correspond to the critical points of the scalar valued (system energy) function

\[ g(u_{t+h}) = \frac{1}{2}(u_{t+h} - w)^T M(u_{t+h} - w) + h^2 E(u_{t+h}), \]  

(3.9)

where \( E \) is a potential energy field such that:

\[ f_{ext} - f_{int}(u_{t+h}) = -\nabla E(u_{t+h}). \]

The task is then to minimise (3.9) to obtain \( u_{t+h} \).

Liu et al. reformulate \( E \) by introducing a set of rest-length spring directions \( d_i \) (that must be found at each simulation step) and setting the potential energy of the system to be the sum of the potential energies of each individual spring. The task is then to optimise for both unknowns \( u_{t+h} \) and \( d \). A block-coordinate descent (or alternating local-global) method is employed to solve for each of \( u_{t+h} \) and \( d \) individually in an alternating, two-step fashion by fixing the other unknown, giving fast and stable results [71].

In our evaluation, we set the time step to \( h = \frac{1}{60} \text{s} \) since the method is implicit (and therefore unconditionally stable) with 10 iterations of the solver per update (which are the default values set by the authors in their work). The same procedure as in Section 3.3.2 is carried out, with signals given in Figure 3.13 together with error metric scores. The scores are excellent for a discretisation of at least \( 17 \times 5 \times 5 \) nodes, with the optimal score being for the \( 29 \times 8 \times 8 \) resolution. Liu et al. acknowledge that, for a swinging curtain scenario, just one iteration yields a “plausible” simulation but lacks particular detail and looks “a bit inflexible”. Whereas for 100 or 1000 iterations of the solver “it is difficult to tell the difference” from their reference simulation. These qualitative observations
are reflected in our quantitative error metric ratings for this test 3D cantilevered beam scenario on a $17 \times 5 \times 5$ mesh:

<table>
<thead>
<tr>
<th>Iterations</th>
<th>$e_{ISO18571}$</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.910</td>
<td>Good</td>
</tr>
<tr>
<td>5</td>
<td>0.984</td>
<td>Excellent</td>
</tr>
<tr>
<td>10</td>
<td>0.984</td>
<td>Excellent</td>
</tr>
<tr>
<td>100</td>
<td>0.984</td>
<td>Excellent</td>
</tr>
<tr>
<td>1000</td>
<td>0.984</td>
<td>Excellent</td>
</tr>
</tbody>
</table>
Figure 3.13: In our framework, the MSS is updated using a block-coordinate descent method as outlined in [71]. **Above:** The bidirectional Hausdorff distance measuring the mesh displacement is plotted against time for the test (MSS) signals at different resolutions together with the reference (NLFEM) signal. **Below:** The ISO18571 is used to score the test signals against the reference signal. Further resolutions are plotted that are absent in the above graph.
3.4 Discussion

The smaller, inner mesh of Section 3.3.2 produces an ISO18571 rating of 0.985, which grades as excellent, yet could be closer still to a maximum score of 1. This value takes into account any signal fluctuations corresponding to unavoidable factors such as numerical error. Such a high score as 0.985 is not an unrealistic expectation since the expressions are derived analytically in the literature by Kot et al. in [68], matching the parameters of MSS to NLFEM. The smaller mesh is offset (see Figure 3.11) and, upon reconstruction to the larger, outer mesh, the fixed surface could exhibit stiffer behaviour than desired (particularly at the fixed face) that in turn influences the results. A larger resolution is unstable in this scenario but in combination with smaller timesteps a better result could be produced. This is a trade-off with performance.

The second MSS model evaluated in Section 3.3.2 obtains the solution at each update step by minimising the energy or the system. The optimisation procedure for doing so can make the solution more approximate to that obtained in the first MSS model. The optimisation problem iteratively solves for a minimal result but is not guaranteed to reach a convergent solution for any number of iterations. We observe a loss of energy in Figure 3.13 because the measured Hausdorff displacement does not return to 0 metres as it does in the reference model. This is due to the implicit backwards Euler integration scheme being energy dissipating, as well as the minimisation performed in the iterative solver not necessarily converging. Nevertheless, the authors set the iteration count to 10 as it offers “the best trade-off between speed and [visual] quality”. If we increase or decrease the number of iterations then generally this increases or decreases the accuracy of the solution, respectively. However, to help to achieve real-time performance in an application, the iteration count cannot be too large. This is one
trade-off that must be leveraged and the effects can be seen directly within the framework in the output scalar value. The framework allows the user to achieve their desired performance goals whilst still having a quantitative grasp on the accuracy of the PBDM being developed. Moreover, we can now see that in these setups, by direct comparison, the local-global iterative solver (with 5 iterations) of Section 3.3.2 is marginally less accurate than the semi-implicit integration scheme of Section 3.3.2: $0.984 < 0.985$. Whereas at just one iteration, the method is less accurate than with 5 iterations: $0.910 < 0.984$.

In the evaluations presented, the Hausdorff distance, popular in computer vision and shape matching, in particular, is used as the error measure at each update step. Assuming no \textit{a priori} knowledge of the deformation, every node must be evaluated at each step. This makes for an $O(n^2)$ algorithm and is not always suitable for real-time applications. Otherwise, prior user knowledge of the specific scenario can allow for fewer nodal comparisons, reducing computational costs. Nevertheless, faster algorithms exist that can reduce the computational expenses (such as the optimised Hausdorff distance algorithm in [72]), should that be necessary, which is possible via a simple change in the displacement measure component.

We chose to set the (very soft) 3D cantilever scenario to run for 5 seconds to allow for a large, unsteady deformation over a substantial period of time whilst not having to wait too long for a fast quantitative evaluation of accuracy. We note that the time to obtain quantitative results depends on how long the scenarios are set to run for. A long scenario time can hinder the user from getting rapid feedback before optimising their test PBDM. Whilst this is not a limitation of the functionality of the framework, it reduces the level of productivity. That said, the 3D cantilever is unsteady and begins to oscillate with a period of around 5 seconds. We are aware that any phase errors (or simulation instabilities) that may arise
after, for example, 60 seconds, are not evaluated in the framework. We think that this is an acceptable trade-off because it is assumed that a deformation that lasts longer than five seconds is not one that is typical in interactive CG applications. It is rare that in IVEs an object is interacted with once and observed for extended periods of time with no further interaction. Nevertheless, more suitable scenario components can and should be substituted into the framework as needed to test this functionality.

Using the displacements from the undeformed positions could possibly give falsely positive error metric results if the error measure is not a signed measure. In the case of two bodies deforming in opposite directions from their identical initial positions before return back to those initial positions, the error measure, being unsigned, would produce similar results. If a signed measure is used instead, such as outlined in [73], the ISO18571 error metric could potentially take this antiphase into account when evaluating the phase error of the signals. As such, more suitable error measure can easily be substituted into the framework to overcome this. This is done in Chapter 6 when the direction of the deformation of the evaluated cloth scenario is significant.

3.5 Summary

In this Chapter we have presented a component-based software framework that allows us to obtain an objective, quantitative evaluation of accuracy of a proposed PBDM in real-time CG. This can also have an applicability to fields outside of real-time CG. The use of a single scalar value for accuracy provides an objective measure in a field that we found to mostly rely on subjective, qualitative validations of accuracy through visual plausibility. The framework is composed of components that are connected in a software architecture, such that each component is a
replaceable modular unit that exposes some necessary functionalities through its interfaces. The implementation details of these components are presented in Chapter 4.

A key component of the software framework is the test PBDM component, which allows the user to test any PBDM within the framework. The test PBDM is employed to simulate a scenario in which its produced nodal displacements are compared with the nodal displacements of a reference deformation at every update step of the simulation over a prescribed time period. We produce a rating that quantifies the level of agreement between the two deformations and each rating is categorised by a preset ISO standard that describes how good the agreement is. The descriptions come from grading the output rating and the grade boundaries are set by experts in vehicle safety, an important engineering field. The grade boundaries are strict and represent the opinions of experts of what it means for a pair of signals to be in “excellent agreement”.

For the evaluations conducted, the reference deformation is provided by what we have established as the “gold standard” in PBDMs. This is the nonlinear finite element method used to solve the equations forming the principle of virtual displacements from elasticity. Two competing PBDMs, which are popular in the CG literature, have been evaluated within the proposed framework, giving scores that allow the direct comparison of PBDMs against one another.

The software framework allows for the fast development of test PBDMs by providing quantitative validation that is not limited by the time it takes to get qualitative validation from a field expert. The fast quantitative validation allows for an immediate, progressive iteration on the development of deformations of a test PBDM to match the chosen reference deformation as close as possible. In Chapter 5, we perform optimisations of model properties specific to PBDMs by trying to increase the accuracy rating provided by the framework. We note that
the framework is not a diagnostic tool to help identify the sources of errors but leave it as future work beyond this thesis. The extension of the framework to the fields of fluid simulation and fracture simulation is possible without a fundamental change in design but is also not something that is presented in this thesis.
Chapter 4

Software Framework for Quantitative Validation

So far we have presented a component-based software framework in Chapter 3 to allow for the quantitative validation of an entire time-evolving deformation history of a PBDM as opposed to a quantitative validation at a single time instance. We term this the *quantitative validation of deformation histories* (QVDH) procedure. Overall, we want the framework to:

- implement the QVDH procedure through the use of interfaces;
- implement the optimisation of model material properties for any PBDM (Chapter 5);
- implement the switching of PBDMs based on some predetermined accuracy requirements (Chapter 6);
- allow for the rapid exploration of the above ideas through its components;
- have reusable and replaceable components so as to be sufficiently general.
Whilst in later Chapters we will discuss PBDM optimisation and PBDM switching, in this Chapter we overview the quantitative validation software framework and discusses the necessary components that combine to make QVDH procedure possible. In Section 3.3.2, we found the quantitatively validated results in our presented framework of the tested PBDM were in agreement with the qualitative conclusions made about the tested PBDM by the original authors. Validation of the deformation history is more thorough than validation at a single point in time, whilst quantitative validation provides objective measures of accuracy in contrast to qualitative validation techniques such as visual plausibility that are subjective to the user. The software framework was first presented in Figure 3.4 but is shown again below for convenience.

The framework is a component-based software framework and, in order to perform the quantitative validation procedure, all of the listed components must be instantiated. The components of the software framework are:

- **Scenario** component;

- **Regulator** component, containing:
  - 2 **Entity** components, containing:
    - **Mesh** component;
    - **Body** component;
* (Physically Based) Deformable Model component(s);

- 2 Discrete Time Signal components;

- Displacement Measure component;

- Error Metric component.

These components interact to transfer data through their necessary interfaces. The flow of data is illustrated in Figure 4.1. Constructing the software framework using components provides convenience to the user through the ability to easily and quickly get quantitative feedback on how accurate a tested deformation is. The ease of use comes in having to only implement the necessary components, which, combined with the generality of performing any evaluation offers flexibility in performing the desired evaluations of accuracy. This is in contrast to previous works that have been specifically tailored to particular case studies. The quickness of use comes in not having to require validation from a subject matter expert, which in practice could be a time consuming process. The framework outputs an objective rating upon completion of the deformation simulations.

**Chapter overview** In Section 4.1, we outline the component construction stage. We present the intended purposes and interfaces of the components of the framework in Section 4.2. We outline the limitations of the framework in Section 4.3.
Figure 4.1: The data flow, represented by the solid black line, happens via function call on the interfaces of the components of the software framework. The input data to both modelled deformable bodies comes from the Scenario component. This data is iteratively edited throughout the simulation by the Deformable Model components, which is then evaluated by the Displacement Measure component to produce a displacement measure for each body. These measures are passed to Discrete Time Signals components until the duration of the simulation - as set by the Scenario component - is reached. The resulting pair of discrete time signals are then rated for similarity by the Error Metric component to give the final output in $[0, 1]$. 
4.1 Component construction

The components are modular, architectural units; they are primary computational elements and data stores of the system with well-defined interfaces that expose their functionalities [74]. The software architecture for the components of the framework is presented in Figure 3.4 as a component diagram, modelled in UML 2.0 [75]. The ports, representing the interfaces, serve as either sinks or sources for receiving or sending data, respectively (these are “sockets” and “lollipops” in UML 2.0), and are connected via assembly or delegation [74]. The components are composed in the design phase of the component life cycle; all component compositions are made in the design stage and persist until the runtime with no further amendments [74]. This means that the connections between components are fixed at runtime and are made through (function calls of) the interfaces. Components can still be substituted at runtime, however, without a change in the functionality of the framework. The substitution of PBDMs as components is addressed in Chapter 6.

4.2 Component interfaces

With the necessary components in place, the comparison of a pair of deformation histories is regulated inside the software framework. The roles of each of the components of the framework is given in this Section. Note that in the following discussion, the same discrete points on a mesh are referred to as both vertices and nodes. Vertices are with respect to the mesh to be rendered, whereas nodes refer to the discrete points on the physics body at which the solution for the current position is sought. Components may be substituted provided that the contracts set forth by the interfaces are abided by. Each component in the framework is
explained and their interfaces given below. There are two Entity components, one for simulating the reference model and one for simulating the test model. The PBDM that is being evaluated is inserted in as the Deformation Model component on the test Entity component.

4.2.1 Scenario

As seen in Figure 4.1, the initial data for the simulation is set by the Scenario component. The Scenario component sets the following properties of the simulation:

- mesh geometry (the undeformed configuration);
- initial conditions;
- boundary conditions;
- external loading;
- simulation duration;
material properties.

The mesh geometry is what is passed to the Mesh component. It is the surface mesh of the continuum to be simulated, which is tessellated with triangles. The surface mesh is typically constructed in a CAD package. This mesh is thus the boundary of the undeformed configuration of the physical body that is simulated inside the Body component. Initial conditions are obtained through a function pointer to get the initial configuration of the physical body. If this pointer is null then the initial configuration is the undeformed configuration. The external loading is also defined here, as a function pointer, for callback from the Deformable Model components. A typical example of this is acceleration due to gravity acting as a body force or a prescribed traction on the surfaces of the bodies. The duration of the scenario is queried when performing displacement measurements and adding it to the discrete time signals.

4.2.2 Regulator

The undeformed configuration that is passed to the Mesh component via the Entity component is used in the physical simulation of the deformation. Initial conditions can be applied to the undeformed nodes, giving the initial nodal configuration of the simulation. The driver of this data exchange is given by the Regulator component. This component governs the quantitative validation process and contains two Entity components and two Discrete Time Signal components. The Regulator component governs the comparison of PBDM data that is obtained at different time steps, as outlined in Section 3.2. Should the time steps differ between compared PBDMs, then the Regulator component interpolates between physical states of the Entity components to perform comparisons at every simulated time step.
4.2 COMPONENT INTERFACES

4.2.3 Entity

An Entity component stores the visual and physical representations of the body being deformed, which are given by the Mesh and Body components, respectfully. The Body component stores the current and previous nodal states. The number of previous states is determined by the numerical integration scheme used (see Section 2.6), but is typically one previous state. The physical simulation runs at a fixed time step, which is to say that the physics will be deterministic; the same simulation will produce the same results (up to numerical error) for the same chosen scenario. To render the physical state of the bodies, the Mesh component must be updated with the current nodal positional data. The rendering stage of the application does not run at a fixed time step but rather a variable time step [7]; a new image is displayed to the screen when all other processes in the application have completed, the time step for which can vary from frame to frame. A consequence of this is that multiple rendering stage updates could happen in-between physics updates, and so the vertices appear to not move because the next physical state has not yet been calculated. To render the current state to screen smoothly, therefore, some linear interpolation is performed between the previous and current states to give a smooth image, which is discussed, along with other solutions, in [7]. The Entity component is responsible for this synchronisation of the output data of the Mesh and Body.

4.2.4 Mesh

The Mesh component receives as input a triangular mesh from the Scenario component that might have no internal vertices. The Mesh component is then responsible for making the (potentially hollow) surface mesh a volumetric mesh by adding vertices to both the interior and surface. This can be achieved using
Delaunay triangulation algorithms, whose goal is to position the vertices so that the resulting triangles (that are formed by connecting neighbouring vertices) are regular [76][47], which removes long, thin scalene triangles. The triangles can be connected to make tetrahedrons. *TetGen* [47] is an example 3D Delaunay triangulator software package that employs this algorithm to produce tetrahedral meshes, as seen in Figure 4.4. As seen in Figure 4.3, the **Mesh** provides the resulting

Figure 4.3: A mesh component provides the undeformed geometry of the simulation.

![Mesh Component Diagram]

Figure 4.4: A surface mesh of a parallelepiped consists of 8 corner vertices. *TetGen* [47] performs a Delaunay triangulation and adds extra vertices to the surface and the interior to make the mesh volumetric.

![Parallelepiped Mesh Diagram]
volumetric computational mesh as well as the edges of the mesh, triangles and (tetrahedral) elements through its interfaces. The mesh provided is the undeformed geometry to be potentially deformed by the Body component. As mentioned in the discussion of the Entity component, the Mesh component requires an up-to-date version of the deformed nodes of the (physical) Body component in order to output the updated deformed mesh to the user.

4.2.5 Body

![Diagram of Body component]

Figure 4.5: A body component maintains the current state of the time-evolving deformation.

A Body component maintains the current state of the time-evolving deformation. The material properties of the body are set by the Scenario component. These properties can be changed at runtime, however, to adjust the force response behaviour of the body. This is what is done in Chapter 5 when fitting the material properties of any tested PBDM to those of the reference PBDM. The Body component stores the nodal positions, velocities and accelerations. These values are updated at every update stage. The Body component calls upon the
attached Deformable Model components to edit the positions of the nodes that were initially received from the Mesh component. The positions are changed in response external loads as set by the Scenario component. However, fixed nodes do not move. Fixed nodes are determined by the boundary conditions, which also are set by the Scenario component. The dynamics of the nodes is determined by the currently attached Deformable Model component and the currently used numerical integration scheme.

4.2.6 Deformable model

A Deformable Model component represents a PBDM and governs the dynamics of the nodes of the body to which it is attached. A body can have any number of Deformable Model components attached to it, but only one is active at any one time. The interfaces of Figure 4.6 provide a means to potentially move the nodes of the body using the underlying algorithms of the PBDM, examples of which are given in Section 2.5. The Deformable Model components require the
initial state, external loads and material properties (given through delegation connections via the Body component from the Scenario component) and the integration scheme used, which can be set using the interface of the Body component. For some deformable models, changing the integration scheme does not have an effect. For example, non-physically based deformable models do not have to satisfy force-balance equations and so avoid numerical integration of these quantities altogether. Nevertheless, the provided Deform interface in Figure 4.6 still is the means to deforming the current pose\(^1\) of the body. This is the case for deformable models based on shape-matching algorithms [27]. The framework does not care for which model is used to deform the body.

Different PBDMs exploit different levels of connectivity of the geometric mesh. For example, a MSS requires edges across which springs can be placed, whereas FEM requires a list of nodes that make up the elements of the mesh. These are accessible through the interface of the Mesh component.

### 4.2.7 Displacement measure

A Displacement Measure component quantifies the level of displacement of the current pose of a body (against its undeformed pose) as a single value. It is a quantification of displacement at a single instance in time. The displacement at each nodal point is the difference between the deformed and undeformed position vectors (from the origin). The means of extension of the measure to a single quantity that expresses the displacement for the entire set of nodal points is what is implemented in (and differentiates) each Displacement Measure component.

At each update stage of the simulation, the deformable bodies may have deformed. Performing a displacement measure at every update stage therefore

---

\(^1\)The pose of a body is the current position and orientation of the nodal points of the body.
Figure 4.7: A displacement measure component measures the displacement of the current state (of the deformed body) at the current pose.

gives a measure of displacement at discrete time instances of the deformation. These measures are performed at every update stage and this data is passed to the corresponding Discrete Time Signal component by the Regulator component.

How the displacements are compared is governed by the Regulator component. For example, each signal could contain displacement measurement data unique to each modelled deformable body, so that there is one independent signal for each deformable body. Alternatively, the difference of the displacement measures can be computed to produce the final displacement value to be added to the signals, so that each discrete time signal can contain displacement measure data from both Body components.

4.2.8 Discrete time signal

A discrete time signal (DTS) is a container whose elements are a pair of floating point values representing some measure of interest and the time at which that measure was taken. The form of the container is determined by the Discrete
4.2. COMPONENT INTERFACES

Figure 4.8: A discrete time signal component stores a time-evolving measure as a container.

Time Signal component and is typically an array. The time-evolving displacement measures of each body are collected from the Displacement Measure component to form a pair of arrays that become a pair of discrete time signals.

4.2.9 Error metric

Figure 4.9: An error metric component compares the agreement of two discrete time signals and produces a single error quantity between 0 and 1.

An Error Metric component takes as input two discrete time signals, which
are passed to it from the Discrete Time Signal components by the Regulator component. An Error Metric component quantifies the level of agreement between the two signals to give an error measure between 0 and 1, where 0 is a perfect match. An example instance of an Error Metric component was in using the ISO18571 error metric in the quantitative validation technique of Section 3.3.

4.3 Limitations

Performing a quantitative validation using the presented software framework requires the storage of large amounts of data. This is necessary because displacements require the data from the undeformed pose as well as the current, deformed pose. The deformed pose is updated at every update step and the computations of each appropriate component is performed, in turn potentially storing more data. A longer-running scenario will store more data. However, this is also a limitation of the works in the literature (see Section 3.1) which (also) evaluate against multiple poses.

Having to simulate a pair of deformations governed by a reference and test PBDM can be a time consuming procedure. The reference PBDM serves as the ground truth because of its accuracy and cannot normally be run in real-time if the reference PBDM is chosen as a computationally expensive PBDM such as NLFEM. If the output of the reference deformation is pre-computed the computational performance is improved at the expense of having to store more data and the scenario being unchangeable. If the reference deformation is computationally expensive and simulated at runtime, therefore, the framework does not run in real-time also.

The advantage of generality in being able to evaluate any scenario brings the disadvantage of requiring more preliminary work in order to have the components
communicate appropriately within the framework. This is worth the effort, however, as only having to implement the necessary interfaces means that the components are reusable.

4.4 Summary

This Chapter presented the software framework architecture used to quantitatively validate a PBDMs deformation by evaluating its entire output deformation history. This is achieved through the use of software components. The components are instantiated before the runtime and require implementation of well-defined interfaces, which were presented in Section 4.2. The components communicate and transfer data at runtime to produce an objective rating of accuracy between $[0, 1]$.

The framework allows for component substitution which makes the framework flexible to use in a wide variety of scenarios and applications. Whilst the software framework is not currently available for public use, we hope that it is simple enough so as to be easy to recreate.
Chapter 5

Optimisation of Model Properties

5.1 Introduction

How a PBDM models the response under external loads is governed by the model properties of the PBDM. The desired softness of the modelled material depends on the purposes of the simulation being performed and is traditionally achieved through the adjustment of its material parameters until the behaviour is satisfactory [2]. The simplicity of this procedure makes it attractive, especially when accuracy is not of great significance and visually plausible animations are desired. If we assume that the material is homogeneous and isotropic, then the behaviour of the material can be governed by just two scalar quantities, which greatly simplifies the material property selection process [5]. For more complex heterogeneous and/or anisotropic materials, more material properties are required. Manual adjustment of model (material) properties is ideally limited to adjusting a handful of parameters rather than adjusting many because it makes our task simpler.

In Chapter 3, we presented a component-based software framework that facilitates the construction of a single scalar value, $q \in [0, 1], 1 = \max$ that
quantifies the level of agreement of two PBDMs’ complete deformation histories. The interfaces of the components of the software framework were given in Chapter 4, with emphasis placed on the fact that the components were easily replaceable to improve the resulting quantitative assessment. This Chapter extends the software framework of Chapters 3 and 4 by using the QVDH procedure of Chapter 3 to optimise model properties for any test PBDM against a reference PBDM that serves to provide ground truth data. As a summary, in Figure 5.1 we present the evolution of the framework from Chapter 1.

**Figure 5.1:** The evolution of the QVDH software framework from Chapter 1 up until here.

Optimising for model properties has been covered before in the CG literature for specific PBDMs. We have found that the PBDMs’ model properties are primarily optimised against real-world data serving as the ground truth at multiple quasi-static states. We discuss this further in Section 5.2. A problem with this approach is that the gathering of real-world data requires specific apparatus such as force-tracking devices or depth-measuring cameras. Moreover, previous works focus on
optimising a single PBDM against ground truth data. Whilst the ideas presented in these works could potentially be applied to further PBDMs, it is not clear what preliminary work is required in order to undertake this. In this thesis, our software framework allows for the model properties of any PBDM to be optimised against the selected ground truth data without the need for any specialised apparatus. Furthermore, we evaluate the entire deformation histories and not just quasi-static states, thus we also evaluate the motion up between each quasi-static state.

We set the objective function for the optimisation process to be $e = 1 - q \in [0, 1]$, so that an error of 0 is minimal as one might expect. The optimisation problem is then a minimisation problem in which we seek the set of model properties for the PBDM being optimised that best minimise the objective function. As we did in Chapter 3, we choose the NLFEM to be the reference PBDM for the problems addressed in this Chapter. We also present the additions that need to be made to the existing software framework to achieve this optimisation. We find that the objective function produces many problematic, local extrema and so we use a global optimisation search strategy, which is implemented in dlib[77]. In Section 5.4, the presented method is shown to converge on an analytical result for a 3D cantilever known a priori for a MSS, which was the result that we used in the results section of Chapter 3 also. The convergence to this analytic solution shows the correctness of the optimisation procedure when using the ISO18571 error metric. We also present the best-fit parameters for PBD models whose model properties do not have a physical grounding (see Section 2.5.6). This shows that our optimisation procedure is able to handle any general deformable model.

The structure of the Chapter is as follows. In Section 5.2 we introduce the related work and background material and compare our proposed optimisation method to the similar works in the literature. In Section 5.3 we formulate the optimisation problem and outline the necessary extensions to the existing QVDH
framework. Results are given in Section 5.4 for 3D cantilever and moving cloth scenarios for two tested PBDMs popular in the literature: MSS and PBD. We follow this with a discussion of the results in Section 5.5. We conclude the Chapter with Section 5.6, where limitations to this work and future work is also discussed.

5.2 Background and related work

Selecting material properties to produce the desirable deformations is a complex procedure that is made more complex if the material is heterogeneous, anisotropic and its response changes depending upon its current pose (for example, a muscle exhibits different material responses when contracted as opposed to when it is stretched). If we make assumptions about the material being a homogeneous, isotropic and hyperelastic one, the behaviour of the material can be governed by two parameters that link the displacements to internal forces [5]. Otherwise, producing a constitutive law to produce the desired behaviour in more complex materials presents a more difficult challenge.

5.2.1 Producing nonlinear constitutive laws from data

Nonlinear constitutive laws for heterogeneous and/or anisotropic materials have been developed by using data captured on real-world objects in multiple poses, which are found in key frames extracted from video sequences. For 3D solids, Bickel et al. [78][79] solved multiple quasi-static problems using CoRotFEM when they applied different surface loads in order to obtain multiple force-displacement samples and thus get a field of stress-strain relations, with their latter work concerning the interaction of layered materials specifically. They capture nonlinearity by varying the parameters depending on the measured (Cauchy) strain and then interpolating over the field in an element-wise manner, which gives a heterogeneous
material response. In a similar fashion, Wang et al. [80] and Miguel et al. [81] use real-world data from cloth set up in some special apparatus together with a piecewise linear elastic model to model nonlinear and anisotropic stretch and bend of cloth. However, it is noted by Miguel et al. in [82] that force-based formulations of a nonlinear constitutive law fail to keep the system conservative. Therefore in their work (and also in the work of Clyde et al. [83]), they produce necessarily conservative systems by using energy-based formulations in which a single hyperelastic energy function is fitted against a subset of input poses as reference data.

5.2.2 Fitting deformations sequences

Some works try to fit a deformation sequence to a priori known (input) data by adjusting different properties of the system and optimising for a sequence that matches the input data well. Li et al. [84] sought both the model properties and the fewest number of externally applied loads necessary to fit a deformation history. Similarly, Xu et al. [85] fit the Young’s modulus (describing the stiffness of the material - see Section A.1) in an element-wise manner. The constraints on the sequence in [84] are positional (corresponding to part of the body being in a certain position at a specified time) and the system does not produce explicit materials properties but rather implicitly provides them through changing parameters relating to the particular animation sequence, which is reflective of a material changing its properties as the pose of the body changes. However, Pan et al. remark in [86] that this results in “severe dynamics errors” for large displacements and fast nodal velocities. The constraints on the sequence in [85] are given as prescribed displacements and external loads, so that (only) the material properties are optimised for. The goal of these methods is to match simulations
to input sequences that could, in the context of our proposed framework, be the simulation of the reference PBDM. Our task would then be to try and match the deformation of a test PBDM to this reference sequence; the reference sequence just serves as the ground truth to compare against.

5.2.3 How optimisation is typically performed

The work most similar to that in this thesis is by Wang et al. in [87]. They optimise the material properties of a CoRotFEM model so that its deformation simulation matches a sequence of point clouds and surface meshes from real-world data. This is as opposed to force-displacement data gathered in the previously cited works. They use an *Xbox One Kinect sensor* [88] as the apparatus for collecting the real-world reference data. We find that the common practice for the optimisation process in the majority of the works cited so far is to perform a back-and-forth process between the following two steps:

- **Forward step:** solve the quasi-(elasto)static linear equation $Ku = f_{\text{ext}}$ (see Appendix A).

- **Update step:** iteratively change the model properties, guided by some evaluation of the gradient of the objective function at the “current best” model property set.

The above procedure stops when the internal forces $Ku$ balance the external forces $f_{\text{ext}}$. In practice, this is when they are sufficiently close in a sum of least-squares sense. In the context of our proposed framework, a sum of least-squares measure of reference and test nodal displacements would be an appropriate choice of displacement measure component. However, it requires that the two discretised meshes have the same undeformed discretisation into nodal points to compare corresponding nodal displacements during the simulation.
Gradient-based optimisation methods are used in the cited works, for which some require the costly evaluation of some gradient measure by computing a finite difference. In our optimisation procedure we avoid this altogether by using a gradient-free search strategy. The optimisation procedure can (and should) “early exit” if the current material property configuration is not producing a deformation history in agreement with the reference deformation history.

5.2.4 Matching PBDM model parameters

We found that both gradient-free and gradient-based optimisation search strategies have been used to optimise PBDM model properties against the deformation produced by other PBDMs. The objective function considers the PBDMs’ poses at a single instance in time only.

Gradient-free search strategies

In an early work, Deussen et al. [89] optimise MSS spring stiffness values to match the steady deformation of the MSS to analytic data with some success. They use a simulated annealing search strategy with an objective function that calculates the standard deviation of the difference in displacement values to converge on results that are approved visually. In contrast, not only do we perform matching for an unsteady problem, but quantitatively validation of the entire deformation history is built into our objective function. Similarly to [89], Lloyd et al. [90] fit MSS stiffness values by comparing the deformation of a MSS to that of a LFEM. We believe\footnote{It is not \textit{entirely} clear which search strategy the authors employ, although they remark that “most attention [from researchers] has been directed towards Evolutionary and Genetic Algorithms”.} they use evolutionary and genetic algorithms (with an objective function using Euclidean distance between nodal displacements) to find a spring
stiffness parameter set that firstly gives a visually matching deformation in a single scenario before being assessed quantitatively to have an end-state mean displacement measure of 0.1cm for a cylindrical body of undeformed radius 1cm and height 2cm. They also derive analytic expressions for the stiffness values and tend to favour that approach more.

**Gradient-based search strategies**

Frank *et al.* [62] use gradient descent to match FEM material properties to what is measured in real-world deformations, with a clear dependence on having a good initial guess since gradient-based methods tend to get stuck in local optima. Rodero *et al.* attempt to equate FEM material properties to PBD model properties by minimising the relative error (given by the Frobenius norm) of the difference in stiffness matrices defined for each of the FEM and PBD in [91]. They arrive at inconclusive results and are suspicious about the selected error measure. They further remark that material property fitting using nodal (displacement) data “must” substantially improve the results of the work. The method of optimum model property selection presented in this Chapter is done using nodal data in the form of displacements from the undeformed positions. We present our optimisation procedure next.

### 5.3 Optimisation

We presented the QVDH procedure in Chapter 3, which produces a score of accuracy, $q \in [0, 1]$, against a reference deformation. A full score of $q = 1$ represents the best possible agreement between tested and reference deformations. In this Section we use $q$ to adjust PBDM model properties until we obtain a maximal score. There is no guarantee that a maximum score of $q = 1$ can be
achieved, however, so the optimisation search strategy will stop when either some rating threshold is met or the maximum number of trial model property sets has been tested. The proposed error metric is defined as:

\[ e = 1 - q \in [0, 1], \]  

(5.1)

which takes a minimum value of 0. We note that we search for the model property set that governs the material response of the body being modelled. We do not optimise for the level of discretisation of the mesh of the tested body and leave this for future work.

5.3.1 Problem statement

We first denote the QVDH procedure as \( f_{QVDH}(p) \), which is summarised in Algorithm 2. We might have the option to reuse the reference DTS for each iteration of the optimisation procedure, which can have very good performance benefits at the expense of the memory requirements of one signal. However, to do this the displacement measure must not compare the two PBDM simulation data directly; each displacement measure must be uniquely applied to a PBDM. For example, we can reuse the reference signal if the displacement measure measures the displacement of the current reference PBDM nodes from the reference PBDM rest configuration, but we cannot reuse the reference signal if the displacement measure measures the displacement of the current test PBDM nodes from the current reference PBDM nodes, for this might be different for differing model properties. The vector \( p \) is the set of the \( N \) model properties of the test PBDM: \( p = (p_0, ..., p_N) \). We optimise against the deformation history produced by a reference PBDM. For every candidate solution, \( p_c \), we obtain a corresponding value \( f_{QVDH}(p_c) = q_c \in [0, 1] \).
We denote the QVDH procedure as $f_{\text{QVDH}}$.

Algorithm 2

1: **procedure** QUANTITATIVE VALIDATION
2: test PBDM ← $p_c$  \(\triangleright\) Tested model (material) properties
3: while simulating do  \(\triangleright\) Some scenario is being simulated
4:  
5: if saved reference signal then advance reference PBDM
6: advance test PBDM
7: measure test and reference body displacements
8: add each displacement measure to its corresponding signal
9: if can save reference signal AND not saved then save reference signal
10: $q \leftarrow$ Rating of test and reference signals
11: return $e \leftarrow 1 - q$  \(\triangleright\) The rating of the deformation is $q$

Our goal is to find the optimum configuration, $p^*$, in the set of candidate solutions $p_c$ such that the objective function in equation (5.1) is minimised:

$$p^* = \min_{p_c}(e_c) = \min_{p_c}(1 - q_c), \text{ where } q_c = f_{\text{QVDH}}(p_c) \in [0, 1]. \quad (5.2)$$

This evaluation to get each $e_c = 1 - q_c$ is one of an arbitrarily length of computation time and depends on the duration of the scenario.

5.3.2 Algorithm

We remarked in Section 5.2.4 that gradient-based search strategies can get stuck in local minima. We therefore use a (global) gradient-free search strategy. Global search strategies terminate after some condition has been met, typically if the objective function satisfies some predefined tolerance or the predefined maximum number of optimisation iteration steps has been exceeded. It is then hopefully the case that the optimised value is sufficiently close to the global optimum value.

For an $N$-dimensional problem, let the lower and upper bounds on the possible
candidates $p_c$ be given by $a$ and $c$, respectively:

$$a = (a_0, ..., a_N)^T, c = (c_0, ..., c_N)^T.$$ 

The optimisation process has two distinct goals:

(i) Find the region containing the global extremum;

(ii) Find the (local and, therefore, global) extremum in this region.

The selection of $a$ and $c$ determines how much time and effort the search strategy wastes searching in regions of the range away from the global minimum. However, the location of the global minimum is typically not known \textit{a priori} so this is an unavoidable cost.

**Global search and local trust region search strategy**

This search strategy is implemented in the dlib software library [77] and alternates between two steps aiming to accomplish the two goals of Section 5.3.2 above. The first step is a global search algorithm based on AdaLIPO [92] that seeks to establish a function, $U(p)$, that forms a lower bound on the objective function, $f(p_c)$: $U(p_c) < f(p_c), \forall p_c \in [a, c]$. $U(p_c)$ is a simpler, piecewise linear non-parametric function that is always a lower bound on the objective function. Furthermore, $U(p)$ becomes more refined as the data set grows whilst searching the region bounded by $[a, c]$. The second step of the search strategy is a \textit{local trust region search} [93]. This locally fits a quadratic model around the best candidate solution found so far in the search to find the minimum in the local neighbourhood of the current minimum. A key benefit of this algorithm, with regards to computational cost reduction due to parallelism, is that it scales linearly due to the iterative refinements made to $U(p)$. This means that at each search
iteration, multiple independent computational tasks can be assigned to multiple computational cores and their results later collected to improve the bound that $U(p)$ forms on the objective function. It is designed to perform as few objective function evaluations as possible [77], making it a suitable global search strategy for our problem.

5.3.3 Extensions to the existing software framework

The initial framework described in Section 4 had components for:

- a scenario;
- a reference PBDM;
- a test PBDM;
- a displacement measure that acts on the nodal data of the PBDMs;
- two discrete time signals (DTSs) that hold the error measure data over time;
- an error metric that acts on the DTSs to give a final resulting score.

To use this software framework in our optimisation problem, we implement additional ports with interfaces on the existing components of the software framework and define new Optimiser, Optimisation Problem and Objective Function components whose interfaces connect with the new interfaces at those ports. See Figure 5.2. The use of these optimisation-related components is optional and the optimisation procedure is only performed if all of these components are implemented.

If the reference DTS measures the displacement of the reference PBDM and the deformation of the reference PBDM is fixed, the reference DTS can be stored from a previous simulation and reused for each error metric valuation. Not having
include Objective Function, Optimiser and Optimisation Problem components, which are highlighted in grey.

Figure 5.2: The UML 2.0 diagram of the component-based software framework described in Chapter 3 is now extended to...
to compute a DTS for the reference PBDM can save significant time and resources during the optimisation procedure.

5.4 Results

In this Section we evaluate the effectiveness of our proposed optimisation procedure. We show the correctness of the optimisation procedure in Section 5.4.2 by showing that it converges on MSS model properties known \textit{a priori} from the literature \cite{68} (we have already used this same model property equivalence in Section 3.3 when we validated the results of the QVDH procedure). We simulate the MSS using two numerical integration schemes: implicit Euler (order accuracy one) and implicit trapezoidal (order accuracy two). We note that the implicit Euler schemes numerically damps the motion (which was a problem that we discussed in Section 2.6) and the implicit trapezoidal scheme does not, and this gives different nodal displacements at each update step. Nevertheless, we find that the optimisation search strategy converges on the same model property set, showing the insensitivity of the optimisation procedure to numerical damping.

5.4.1 Experimental setup

We choose NLFEM as the reference PBDM in the following experiments, as we did in Chapter 3. The NLFEM code is once again from \texttt{oomph-lib} \cite{42}. The displacement measure is the bidirectional Hausdorff distance from the undeformed nodal positions to the current (deformed) nodal positions. This allows for the minimisation of the difference in displacements over time. The displacements are measured at every time step, producing a separate signal for both reference and test PBDMs and their similarity is rated using the ISO18571 metric to give a rating $q$. We then use this quantity in the objective function $e = (1 - q)$, which
we minimise in order to find the optimal model property sets. We optimise the following PBDM model property sets for 3D cantilever and cloth scenarios.

• 3D cantilever
  
  − MSS with one spring stiffness. This shows the correctness of our optimisation procedure because the optimal solution agrees with an analytic solution.
  
  − MSS with two spring stiffnesses. The solutions are also in agreement with analytic results.
  
  − MSS integrated using implicit backward Euler and implicit trapezoidal integration schemes to show the insensitivity of the method to the artificial numerical damping of implicit backward Euler.
  
  − PBD optimising the iteration count of the constraint solver, $n_s$, to show its effect on the stiffness of the modelled material.
  
  − PBD optimising the distance constraint stiffness and $n_s$.
  
  − XPBD optimising for $n_s$.

• cloth
  
  − PBD optimising for $n_s$ and two constraint stiffness values for distance and shearing.

We give the results of the 3D cantilever beam scenario in Section 5.4.2) and the moving cloth scenario in Section 5.4.3. We found that both of these scenarios are popular in the CG literature when new PBDMs are developed and introduced. Each scenario models a homogeneous, isotropic material.
5.4. RESULTS

5.4.2 Scenario 1: 3D cantilever beam

This scenario is a 3D cantilevered beam that deforms under gravity, which is the same scenario as used previously in Section 3.3. The resolution of the mesh is selected such that finer meshes do not improve the accuracy of the reference NLFEM model. In other words, discretisation error is minimised for the reference PBDM. The scenario is 5 seconds in duration. The cantilever oscillates multiple times and we do not model damping in the system. Any resulting damping experienced during the simulation of a tested PBDM is a result of its solution procedure. We first show the effectiveness of the optimisation procedure by showing that it converges on an analytic result.

Correctness of the optimisation method: matching a MSS model to an analytic result

One spring stiffness value  The model properties of a MSS are the stiffness coefficients of the springs: \( k_i \). In this experiment, all springs share a single stiffness coefficient \( k \). In the reference NLFEM model, the material has Young’s modulus \( E = 4000 \text{Nm}^{-2} \) and Poisson’s ratio \( \nu = \frac{1}{4} \). The modelled cantilever beam is \( 4m \times 1m \times 1m \) and is discretised with \( 25 \times 7 \times 7 \) nodes. The NLFEM reference curve of bidirectional Hausdorff distance against time (as also done in Section 3.3) is given as the solid line in Figure 5.3.

As explained in Section 3.3, for a mesh with cubic unit cells (or elements), it can be shown that \( k^* \), the ideal spring stiffness, is \( k^* = \frac{2}{5} Ea \), where \( a \) is the length of each unit cell [68]. A \( 25 \times 7 \times 7 \) mesh has \( a = \frac{1}{5} m \) and so \( k^* = 266\frac{2}{3} \text{Nm}^{-1} \). The MSS model with this spring stiffness produces the test signal seen in Figure 5.3. Of note is the clear damping of the displacements, which is due to the implicit integration scheme being of order accuracy one. As such, the error for
$k = k^*$ is $e = 0.15$. The upshot of this is that we have a ground truth, analytic

\[
\begin{align*}
15. \text{The upshot of this is that we have a ground truth, analytic} \\
\end{align*}
\]

\[
\begin{align*}
\text{Figure 5.3: The 3D cantilever scenario with the FEM (solid) serving as the} \\
\text{reference. The tested MSS (dashed) with analytic result } k = k^* = 266\frac{2}{3} \text{Nm}^{-1} \text{ is} \\
\text{also plotted. Additionally, } k = 1000 \text{Nm}^{-1} \text{ is plotted (dotted) to show the effect} \\
\text{that an overly large stiffness has on the measured displacement.} \\
\end{align*}
\]

\[
\begin{align*}
\text{answer that we can compare our optimal model property against. The bounds} \\
\text{of the optimisation problem are } k \in [1, 4000] \text{Nm}^{-1}. \text{The optimisation procedure} \\
\text{produces the data in Figure 5.4. The initial spring stiffness is selected at random} \\
\text{between the bounds. The minimum is achieved at } k = 266.56 \text{Nm}^{-1} \text{ with error} \\
e = 0.152, \text{which is very close to the analytic solution } k^* = 266\frac{2}{3} \text{Nm}^{-1}. \text{We observe} \\
\text{many local minima scattered at values around } k_g = 540 \text{Nm}^{-1} \text{ and greater. A local} \\
\text{search strategy such as gradient descent would converge towards one of these local} \\
\text{minima had the initial guess been greater than } k_g. \text{This shows the dependence} \\
\text{that local search strategies have on a good initial guess in the presence of many} \\
\text{local minima.}
\end{align*}
\]
5.4. RESULTS

The 3D cantilever scenario is modelled using a MSS model with one spring stiffness. The best fit parameter is achieved at the minimum point \( \{k = 266.56\text{Nm}^{-1}, e = 0.152\} \).

Two spring stiffness values  The simplicity of selecting a single spring stiffness value to model material response is an appealing prospect. However, as we have seen materials response is often determined by multiple material property values. To show the effectiveness of our optimisation procedure, we optimise two spring stiffness values \( k_1, k_2 \). Of course, setting \( k_1 = k_2 = k^* \) will produce the same optimum results as in the single spring stiffness case. However, a general relation between the Young’s modulus \( E \) and any number of stiffness values \( k_i \) acting on springs of rest length \( L_i \) is shown in [68] to be:

\[
E = \frac{1}{6V} \sum_i k_i L_i^2.
\]  

(5.3)

There are many variations to the distribution of the two spring stiffness values to springs. Suppose we have \( m \) spring constraints that are stored in an array so that each has a unique spring index, \( s_i \in [1,m] \). We test three distributions of the
spring stiffness values $k_1, k_2$ to the spring with index $i$ that has a spring stiffness value $k_i$.

1. Half-and-half:
   
   $$ k_i = \begin{cases} 
   k_1 & \text{if } i < \frac{m}{2} \\
   k_2 & \text{otherwise.} 
   \end{cases} $$

2. Alternating:
   
   $$ k_i = \begin{cases} 
   k_1 & \text{if } i \text{ is odd} \\
   k_2 & \text{if } i \text{ is even.} 
   \end{cases} $$

3. Random: each $k_i$ is selected from $\{k_1, k_2\}$ at random.

The plots for each of these distributions are are illustrated as heat maps in Figure 5.5. We can see that minima are densely populated around the solution $k_i = k^* = 266\frac{2}{3} \text{Nm}^{-1}$, which shows that multiple candidate model property sets satisfy equation (5.3). We have summarised the weighted average values in Table 5.1 and note that in each case, the weighted average was close to $k = k^*$. We have shown that our proposed optimisation procedure effectively finds minimising model property sets. Next we conduct the optimisation for MSS using two different numerical integration schemes. The first is an implicit Euler that numerically damps the motion. The second is an implicit trapezoidal method that does not.

### Table 5.1

<table>
<thead>
<tr>
<th>Distribution</th>
<th>$k_1$ (Nm$^{-1}$)</th>
<th>$k_2$ (Nm$^{-1}$)</th>
<th>Weighted average (Nm$^{-1}$)</th>
<th>$e$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Half-and-half</td>
<td>299.2</td>
<td>233.8</td>
<td>266.5</td>
<td>0.0154</td>
</tr>
<tr>
<td>Alternating</td>
<td>252.9</td>
<td>275.4</td>
<td>264.2</td>
<td>0.0151</td>
</tr>
<tr>
<td>Random</td>
<td>304.6</td>
<td>224.7</td>
<td>264.1</td>
<td>0.0153</td>
</tr>
</tbody>
</table>

| Table 5.1: This table shows the weighted average values (as per equation (5.3)) of the optimal spring stiffness values $k_1, k_2$ for the labelled distributions within the MSS model. The averages are within 2.5Nm$^{-1}$ of the analytic result, $k^* = 266\frac{2}{3}$Nm$^{-1}$. |
Figure 5.5: The 3D cantilever scenario is modelled using a MSS with two spring stiffness values. The stiffness values are distributed in three distributions and plotted as heat maps as follows. Top left: half-and-half, top right: alternating, bottom: random. The best fitting stiffness values are achieved around the minimum point ($k_1 = k^*$, $k_2 = k^*$) and are shown as the darker regions of the maps. Whilst the minimising values are not exactly $k_1 = k_2 = k^*$, their weighted average is approximately $k^*$ (see Table 5.1). The minimising values for each distribution are circled in red.
Nevertheless, the optimal spring stiffness value that we get is approximately the same.

Mass-spring system: integrated using implicit methods

We set the elastic modulus to $E = 7500\text{Nm}^{-2}$ and Poisson’s ratio is kept at $\nu = \frac{1}{4}$. We also now discretise the mesh into tetrahedral elements using the tetrahedralisation procedure introduced in Section 4.2.4 instead of cubic elements\(^2\). The optimisation of a single stiffness $k$, used for all springs, is illustrated in Figure 5.6 for an implicit backward Euler method. It is illustrated in Figure 5.7 for an implicit trapezoidal method.

Figure 5.6: The 3D cantilever scenario is discretised into tetrahedral elements. The MSS models this mesh by inserting springs along the edges of the mesh (which come from the tetrahedral elements), all with the same stiffness. The best fit parameter is achieved at the minimum point $k = 780\text{Nm}^{-1}$ with error $e = 0.188$.

\(^2\)Discretising the mesh into tetrahedral elements means that the analytical formula of Section 5.4.2 no longer holds.
5.4. RESULTS

Figure 5.7: The MSS from Figure 5.6 is now numerically integrated using an implicit trapezoidal method, which is a second order accurate numerical integration scheme. The best fit parameter is achieved at the minimum point $k = 782 \text{Nm}^{-1}$ with error $e = 0.008$.

The implicit backward Euler integration scheme has order accuracy one and artificially damps the motion by dissipating energy as discussed in Section 2.6. The implicit trapezoidal method has a larger order of accuracy of two and does not dissipate the energy because it is a symplectic scheme. In Figure 5.8 we plot the signals for each MSS model. We can clearly observe the energy dissipation of the implicit Euler scheme. Nevertheless, a minimising stiffness of $k = 780 \text{Nm}^{-1}$ is obtained with error $e = 0.188$. In the case of the implicit trapezoidal method, we obtain $k = 782 \text{Nm}^{-1}$ with error $e = 0.008$, which is a large accuracy gain compared to the implicit Euler scheme.
Figure 5.8: The tested MSS with a single stiffness $k = 780\text{Nm}^{-1}$ is modelled using implicit Euler (dashed line) and implicit trapezoidal (dotted line) numerical integration schemes. We plot it against the reference NLFEM (solid line) with $E = 7500\text{Nm}^{-2}, \nu = 0.25$. Note the damping of the signal of the tested MSS using implicit Euler due to it being first order accurate. The error produced by the QVDH procedure is $e = 0.188$. The implicit trapezoidal method is more accurate, with $e = 0.008$.

Position-based dynamics model: optimising the iteration count

In this experiment we optimise a position-based dynamics model that uses distance and volume constraints. The distance constraints are analogous to the (linear) springs in an MSS and the volume constraints serve to keep the material incompressible. As we mentioned in Section 2.5.6, the stiffness of the material is dependent not only on the stiffness of the constraints, $k$, but also the time step, $h$, and the number of iterations of the constraint solver, $n_s$. To show the effect of the iteration count $n_s$ on the stiffness of the material, we fix the time step to update the simulation every $h = \frac{1}{60}$ seconds. We also fix the stiffness to $k = 1$. 
### BDF1 integration scheme

We integrate the simulation forwards in time using a variational implicit Euler (or BDF1) method [32]. The resulting optimisation of $n_s$ is illustrated in Figure 5.9. The minimum is obtained for $n_s = 46$ iterations with $e = 0.189$.

![Constraint solver iterations](image)

**Figure 5.9:** The tested PBD model is numerically integrated using a variational implicit Euler method. The best fit iteration count is achieved at the minimum point $n_s = 46$ with error $e = 0.189$.

For more iterations, the material is too stiff in comparison to the reference signal. For fewer iterations, the material is not stiff enough. As in the case for the MSS integrated using implicit Euler, the energy dissipation arising from the implicit method is visible in Figure 5.10.
Figure 5.10: The tested PBD model is numerically integrated using an implicit Euler method with $k = 1, n_s = 46, h = \frac{1}{60}$ to produce a deformation with error $e = 0.189$.

**BDF2 integration scheme**  We perform the very same PBD optimisation but instead integrate forwards in time using a BDF2 implicit integration scheme, which is a second order accurate multistep method. The results are plotted in Figure 5.11. The minimum is obtained for $n_s = 21$ iterations, but the error is smaller at $e = 0.011$ due to the increased order of accuracy of the integration scheme. In comparison to using the order accuracy one integration method, whilst more computations must be performed per numerical approximation made, fewer solver iterations are performed in order to get the optimum behaviour.
Position-based dynamics model: optimising distance constraint stiffness and iteration count

We next optimise the model properties \( \{k, n_s\} \) of the PBD model (integrated using BDF2). The results are shown in Figure 5.12. The minimum is obtained at \( k = 0.56, n_s = 48 \), which gives a minimising error of \( e = 0.01 \). This is a slight accuracy improvement over Figure 5.11.

Figure 5.11: The tested PBD model is numerically integrated using a BDF2 method. The best fit iteration count is achieved at the minimum point \( n_s = 21 \) with error \( e = 0.011 \).
Figure 5.12: The tested PBD model is numerically integrated using a BDF2 method. The best fit iteration count is achieved at the minimum point $n_s = 48, k = 0.56$, giving an error of $e = 0.01$. There are many other local minima found throughout the optimisation space that have similar error to that at this local minima, which we are visible in the darker regions around the minimum point.

**XPB**D model: optimising iteration count

For the extended PBD (XPBD) model, we select values for $n_s$ and $h$ as in PBD, but also select a value for the compliance (see Section 2.5.6), $\alpha$, which relates to the stiffness as $\alpha = \frac{1}{\alpha}$. In particular, since the reference model has *engineering stiffness* given by the Young’s modulus $E = 7500\text{Nm}^{-2}$, the compliance is roughly given by $\alpha = \frac{1}{7500}\text{m}^2\text{N}^{-1}$. Thus one degree of freedom (the stiffness of the material) in the optimisation procedure is removed and the optimisation is performed (only) on $n_s$ again. The results are plotted in Figure 5.13 for an XPBD method updated using BDF2 producing an minimal error $e = 0.038$ at $n_s = 21$. This error is
marginally larger than the PBD case, however there is a connection to the material properties of the reference NLFEM model via the single compliance value of the distance constraints.

Figure 5.13: The tested XPBD model with compliance $\alpha = \frac{1}{7500} M^2 N^{-1}$ is numerically integrated using a BDF2 method. The best fit iteration count is achieved at the minimum point $n_s = 21$ giving an error of $e = 0.038$.

**Summary**

In this Section we have optimised many different variations of MSS and PBD models. We summarise the optimal values in Table 5.2. Finally, we plot the signals for each in Figure 5.14.
<table>
<thead>
<tr>
<th>PBDM</th>
<th>Integration scheme</th>
<th>Material properties</th>
<th>Error, $e$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSS</td>
<td>Implicit Euler</td>
<td>$k = 779.4$</td>
<td>0.188</td>
</tr>
<tr>
<td>MSS</td>
<td>Implicit trapezoidal</td>
<td>$k = 782.3$</td>
<td>0.008</td>
</tr>
<tr>
<td>PBD</td>
<td>BDF1</td>
<td>$k = 1, n_s = 46$</td>
<td>0.189</td>
</tr>
<tr>
<td>PBD</td>
<td>BDF2</td>
<td>$k = 1, n_s = 21$</td>
<td>0.011</td>
</tr>
<tr>
<td>PBD</td>
<td>BDF2</td>
<td>$k = 0.56, n_s = 48$</td>
<td>0.010</td>
</tr>
<tr>
<td>XPBD</td>
<td>BDF2</td>
<td>$\alpha = \frac{1}{7500}, n_s = 21$</td>
<td>0.038</td>
</tr>
</tbody>
</table>

Table 5.2: This table shows the optimised material properties for each evaluated PBDM against a reference NLFEM with material properties $E = 700 \text{Nm}^{-2}, \nu = \frac{1}{4}$ and the error $e$ obtained for those optimum material properties. For MSS, $k$ is the spring stiffness and has units $\text{Nm}^{-1}$. For PBD, $k$ is the distance constraint stiffness and has no units, whilst $n_s$ is the number of iterations of the constraint solver performed. For XPBD, $\alpha$ is the compliance and has units $\text{m}^2\text{N}^{-1}$.

Figure 5.14: We plot the signal of the different tested PBDMs in this Section against the reference NLFEM model (solid line) for the simulation of the 3D cantilever scenario. The MSS using implicit Euler and PBD using BDF1 both exhibit artificial numerical damping, whilst the MSS using implicit trapezoidal does not. PBD using BDF2 damps the motion slightly over the 5 second scenario.
5.4. RESULTS

5.4.3 Scenario 2: cloth

We model the problem of simulating the deformation of a thin, 3D cloth over 5 seconds. It is of volume $5m \times 5m \times 0.001m$ and is discretised with $26 \times 26$ nodes. Its material properties are $E = 700N/m^2$, $\nu = \frac{1}{3}$. The cloth is fixed at each of the two upper corners and is initially positioned in the $xy$-plane and then released, from which point it falls under gravity. See Figure 5.15. Since the initial conditions take the cloth away from its undeformed state (which is coplanar to the $xz$-plane), the cloth exhibits transverse plane displacements during its deformation.

Figure 5.15: The cloth does not resist bending when it deforms using two PBDMs. These are out-of-plane displacements and an unsigned displacement measure, such as the bidirectional Hausdorff distance, would not distinguish between the direction towards which the cloth is bending. Also, note that $z$ is "down".

This effect of this is that the cloth folds due to its poor resistance to bending, as is visible in Figure 5.15. An unsigned distance measure such as the bidirectional Hausdorff distance used previously does not distinguish between the out-of-plane directions of the displacement vectors. This was not a problem in the cantilever scenario since the beam only ever displaced downwards due to gravity. We use a signed bidirectional Hausdorff distance measure to measure the total displacement.
CHAPTER 5. OPTIMISATION

of the body. This requires the simple extension of obtaining the sign of the measure using one of the methods laid out in [94]. Let the vector from the point \( \mathbf{X}_r \) on body \( A \) closest to the point \( \mathbf{X}_c \) on the body \( B \) be \( \mathbf{c} = \mathbf{X}_c - \mathbf{X}_r \). Let the normal to the surface of \( A \) at \( \mathbf{X}_r \) be \( \mathbf{n}_r \), then we have the signed distance:

\[
d_{\text{SBDH}} = \begin{cases} 
  d_{\text{BDH}} & \text{if } \mathbf{n}_r \cdot \mathbf{c} \geq 0 \\
  -d_{\text{BDH}} & \text{otherwise.}
\end{cases}
\]

Here “BDH” refers to the bidirectional Hausdorff measure as introduced in Chapter 3, and the “SBDH” is the signed version, for which we have \( d_{\text{BDH}} = |d_{\text{SBDH}}| \).

Finally, we note that due to the large transverse-plane deformations, we must use unconditionally stable schemes throughout the following experiments.

**Cloth using PBD**

We optimize a PBD model with the following three model properties.

- Distance constraint stiffness: \( k_e \in [0, 1] \): to keep the body from stretching.
- Shear constraint stiffness, \( k_d \in [0, 1] \): to keep the body from folding.
- Solver iterations, \( n_s \in [1, 100] \).

**MSS as reference PBDM** In this experiment we set the MSS as the reference PBDM. Ostoja [95] gives a relation of stiffness values of a MSS to a material with Young’s modulus \( E \) with Poisson’s ratio \( \nu = \frac{1}{3} \) as

\[
k = Et \frac{\sqrt{3}}{4},
\]

where \( t \) is the thickness of the cloth. This formula is referenced also in [68][90]. We note that this is only valid for in-plane isotropic behaviour when \( \nu = \frac{1}{3} \) and the
mesh is discretised into a hexagonal lattice: six triangular elements meet at one common centre point to create a hexagonal unit cell [95]. If springs lie across each edge of each hexagon, then the relationship between \( k \) and \( E, \nu = \frac{1}{3} \) is given in (5.4) but we reiterate that this is for in-plane strain only. The cloth deformation that we model involves transverse plane displacements also and so the formula does not hold. Therefore we instead model the cloth using rectangular elements with springs across each element edge and face diagonal. These are also referred to as stretch (resistant) springs and shear (resistant) springs in the literature, respectively, so we shall do the same. There are two total stiffness values in the MSS: one for every stretch spring, \( k_e \), and one for every shear spring, \( k_d \). See Figure 5.16. We set the Young’s modulus to \( E = 7000 \text{Nm}^{-2} \) and Poisson’s ratio

![Figure 5.16: We discretise the cloth into rectangular elements. Here we have shown a simple 4 × 4 nodal discretisation. On the right is a rectangular element. It has “stretch springs” along its edges (dashed lines) and “shear springs” along its face diagonals (dotted lines).](image)
\( \nu = \frac{1}{3} \). Lloyd et al. [90] derive the following equations for square elements:

\[
\begin{align*}
  k_e &= \sum_e \frac{5}{16} t E = \sum_e 4.375 \text{Nm}^{-1} \\
  k_d &= \frac{7}{16} t E = 6.125 \text{Nm}^{-1},
\end{align*}
\]  

(5.5)

where the stiffness value \( k_e \) is summed over the number of elements that share the edge spanned by that spring. The cloth has thickness \( t = 0.2 \text{mm} \). Once again, the stiffness relations in equations (5.5) are derived for an elastostatic problem undergoing in-plane strain and will not necessarily be valid for the deformation we use here. We choose the spring stiffness parameters of the MSS model to be based on this nonetheless. Whilst in general it is important for a reference PBDM to be (verified as being) accurate in order to be a reliable reference, we can we still show how the optimisation algorithm can be used regardless. The results of the optimisation are:

\[
\begin{align*}
  k_e &= 0.48, \quad k_d = 0.91, \quad n_s = 37,
\end{align*}
\]

with error \( e = 0.087 \). The resulting signal for this model property set is given in Figure 5.17 along with the MSS reference signal.
5.5. DISCUSSION

The cloth scenario is modelled using a PBD model with the optimised values $k_e = 0.48$, $k_d = 0.91$, $n_s = 37$. It is plotted against the reference MSS signal for a material with $E = 7000\text{N}m^{-2}$, $\nu = \frac{1}{3}$.

Figure 5.17: The cloth scenario is modelled using a PBD model with the optimised values $k_e = 0.48$, $k_d = 0.91$, $n_s = 37$. It is plotted against the reference MSS signal for a material with $E = 7000\text{N}m^{-2}$, $\nu = \frac{1}{3}$.

5.5 Discussion

The effectiveness of the ISO18571 metric  We find that we do not obtain maximum error for very stiff materials (for instance, in the 3D cantilever scenario), even though they deform much less. For example, in Figure 5.11 the PBD model is very stiff when the maximum number of iteration counts $n_s$, exceeds 30. However, the error fluctuates within $0.5 < e < 0.75$ as $n_s \to 60$ despite the PBD model not visually matching the amplitudes of the reference NLFEM model. This is due to the ISO18571 error metric that is used to assess the agreement of the signals. Recall that this metric assesses four features as a weighted linear combination:

- a corridor rating, which measures the physical “closeness” of the signals, with weight 0.4;
- how the maximum magnitudes differ, with weight 0.2;
- how in-phase the signals are, with weight 0.2, and
• how similar the gradients are at each discrete point, with weight 0.2.

The idea behind these sub-ratings is that, for instance, the measure of phase difference does not suffer due to the discrepancies in the magnitude. In this case, the error arising from an incorrect oscillatory amplitude produced by the PBD model will only limit the total error $e$ by its weighting of 0.2. The corridor rating is the weightiest sub-rating of the four, and can be influenced by the other three sub-ratings. For example, if the magnitude is largely mismatched between the reference and test signals, not only will the magnitude sub-rating suffer but the corridor rating will suffer as a result also. A side-effect of this is that, whilst searching the optimisation space, the optimisation algorithm will produce many local minima. These minima are caused by the four independent error features each rating differently allowing for more variation in the final error $e$. When we optimised the PBD for the 3D cantilever, we first used BDF1 and then BDF2. The BDF1 scheme numerically damps the motion more than the BDF2 scheme, and so the magnitude sub-rating is lower than is the BDF2 case. However, the optimal value is obtained for $n_s = 46$ iterations for BDF1 and $n_s = 21$ for BDF2. A larger $n_s$ corresponds to a stiffer material response, and so the material of the BDF1 model is stiffer than BDF2 as well as already being numerically damped more. By looking at Figure 5.14, we can see that the BDF1 model oscillates more in phase with the reference NLFEM model than the BDF2 model, suggesting that the phase sub-rating is larger for the BDF1 model. It is this kind of sub-rating trade-off that produces many local minima in the objective function. Finally, whilst the appeal of using the ISO18571 error metric is in its generality for any pair of signals, it might not be the most suitable error metric for capturing the subtle deformations of the cloth. The upshot of this is that the error metric should be substituted for something more appropriate for use within the software.
5.5. DISCUSSION

Damping of motion  If we consider a scenario such as that in Figure 5.3, we can see that the exact MSS result when \( k = k^* \) has severely damped motion. As such, after the first oscillation, the maximum displacement is never again reached. The ISO18571 metric only considers the points along the signal and has no knowledge about motion being damped; it merely rates how similar the signals are. For such damped motion, we can imagine that a softer material - corresponding to a smaller value of \( k \) - would initially overshoot the maximum amplitude of the reference NLFEM signal but would later have a more agreeable amplitude once the effect of numerical damping has surfaced. In the case of Figure 5.3, a smaller \( k \) causes the cantilever beam to deform too far downwards towards the \( z \) axis, at which point the MSS model fails to conserve volume and the simulation does not recover. This is why the results of Figure 5.4 show large errors for values \( k < k^* \).

Nevertheless, we can construct such an “overshooting” scenario by taking a stiffer spring constant, \( k = 780 \text{Nm}^{-1} \), as the exact result corresponding to a reference NLFEM model with \( E = 7500 \text{Nm}^{-2}, \nu = 0.25 \). If we choose a smaller \( k_o = 650 \text{Nm}^{-1} \), representing a softer material, then we get deformations which produce the signals in Figure 5.18. The exact solution has an error of \( e = 0.131 \) and the overshot solution has error \( e = 0.103 \), which is an improvement. This is a good example showing that the ISO18571 metric considers only the signals that are input to its calculation and the smaller error is a consequence of more of the signal lying within the corridors (from the \( e_C \) sub-rating of the ISO18571 metric) of the reference signal. From this, we can conclude that the results of the ISO18571 error metric in this context depend on the length of the simulation, the amount of damping present, the material properties chosen and the displacement...
Global optimisation  The optimisation search strategy finds many local minima in the objective function by producing (often marginally) smaller errors $e$ in the neighbourhood of current minima. In Figure 5.11 we can see a two local minima beyond $n_s > 30$ iterations. However, our optimisation method is sensitive to this through the necessary use of a gradient-free global search strategy. We do not need to approximate the derivative at each step e.g. through finite differences.
Search strategy  Global search strategies are not guaranteed to find the global minimum and typically terminate once a prescribed number of iterations have happened. Selecting a larger number of iterations of the global search will provide a more thorough search of the search space. The global search strategy that we have used alternates between:

- searching for the minimum value in the neighbourhood of the current-best minimum, and

- searching the rest of the optimisation space for better minima (ideally the global minimum, of course).

There is no guarantee that the global minimum is found, and performing a more thorough search will increase the chances of finding it at the cost of more computation time. For applications that only require visually plausible simulations, however, the search need not be so thorough. This is especially relevant for optimisation problems of multiple material properties. In optimising the iteration count, $n_s$, and distance constraint stiffness, $k$, of PBD integrated using BDF2 for the 3D cantilever beam scenario, many well-scoring minima are found as seen in Figure 5.12, all of which produce excellent scores of accuracy. For performance reasons we would choose the model property set that contains the fewest constraint solver iterations. The error at $n_s = 21, k = 1$ is $e = 0.011$. The optimisation strategy searched in the local neighbourhood of $n_s = 21, k = 1$ and found this to be the local minimum. We can use the different scenarios to evaluate the features of the QVDH procedure.

3D cantilever  The 3D cantilever scenario exhibits a deformation in which the body recovers all of its elastic potential energy without dissipation. As we can be seen by the NLFEM reference signal in Figure 5.3, the total displacement of
the body (expressed as the bidirectional Hausdorff distance from the positions in the undeformed configuration to the deformed configuration) against time is periodic with period 1.6 seconds, with an approximately constant amplitude during each oscillation of 1.78 metres. There is no damping in the reference simulation. As such, the energy conservation of a tested PBDM can be assessed: can the tested PBDM consistently reproduce the same levels of deflection? If so, does the QVDH procedure recognise this effectively? PBDMs integrated using implicit integration schemes of order accuracy one, for instance implicit backward Euler, do not conserve energy. Figure 5.3 shows that the energy is dissipated over time. The corresponding ISO18571 rating is $q = 0.85$, which is categorised as “good” by ISO18571 standards. Despite this value not being a maximum score of 1, the optimisation procedure converges on the (a priori known) exact solution nevertheless, showing the effectiveness of the method when using the error metric $e = 1 - q$. The extension to 2 spring stiffness values also converges around the analytic solutions in accordance with equation (5.3). In each case, the weighted average of optimal stiffness values was within 2.5Nm$^{-1}$, or 1%, of the analytic result for one value as in (5.3), which is consistent with the equivalence relations made in [68] between $k$ and $E$.

**Cloth** We can see in Figure 5.17 that the signed bidirectional Hausdorff distance records the relative direction in which the most displaced node is displaced. There are many bends and wrinkles throughout the simulation, which is visible in, for instance, Figure 5.15. It is through this wrinkling and bending that we get the rapid sign changes in the signals of Figure 5.17. We use a MSS model as the reference and show excellent convergence of a PBD model to it, giving an error of $e = 0.087$. Our presented optimisation method can be used for any scenarios for which we can have a reference PBDM to test against.
5.5. DISCUSSION

**PBD remarks** It is an acknowledged problem that because PBD assumes infinitely stiff energy potentials, the number of iterations of the constraint solver contributes to the stiffness of the system [48]. This is observable in Figure 5.11 in which the stiffness of the distance constraint is fixed to be $k = 1$ and so changing the number of iterations directly changes the error. The optimal value is $n_s = 21$ iterations. Allowing instead for the distance constraint stiffness, $k$, to be optimised as well as $n_s$, the optimum value is found to be at $n_s = 48, k = 0.56$. It can be concluded that any stiffness “lost” by reducing $k : 1.0 \rightarrow 0.56$ is “gained” by increasing the solver iteration count $n_s : 21 \rightarrow 48$. However, more iterations of the constraint solver requires more computational time, so we should select the model property set with a smaller $n_s$ to get better levels of computational performance. For the XPBD model, the number of iterations is successfully optimised for and is in agreement with PBD at $n_s = 21$. XPBD has the added advantage of being physically linked through to the material properties of the reference NLFEM model, which will be of use in Chapter 6 when we mix models.

5.5.1 Limitations and future work

The limitations of this work arise due to the nature of the global optimisation strategy and the selection of displacement measure. Firstly, the optimisation procedure is slow and not suitable for real-time applications. It should be used as a pre-processing step or run in the background of the main application. Secondly, there is no guarantee that the global optimum is found. Choosing a larger number of iteration steps tends to produce a better result but this adds yet more time to the procedure. Thirdly, the search performs poorly for problems which have:

1. a large search space;

2. not enough iterations of the search strategy;
3. many degrees of freedom, or parameters.

The last of these points is something that has been highlighted as problematic in previous works [90], with doubts cast over whether or not the referenced optimisation strategies would be able to handle large numbers of material properties. However, it is believed that the attractiveness of the tested PBDMs comes from the simplicity of having fewer than 5 material properties to adjust. Nevertheless, more steps of the global optimisation strategy can be undertaken to better converge on the global minimum.

The selection of the displacement measure component should be carefully chosen so that the produced measure is useful in producing a discrete time signal. In the experiments carried out in this Chapter, both bidirectional Hausdorff distance (BDH) measures were used, that give a single scalar measure of overlap of two sets of nodal positions. The unsigned measure has the slight problem of not distinguishing between which set of nodal data lies outside the other. This is not found to be an issue in the 3D cantilever scenario but the signed BDH distance should be used in general. A problem with the signed BDH distance is that it is does not give an intuitive measure at each step, and in cases where the sign is rapidly fluctuating (e.g. in the cloth scenario, Figure 5.17), the calculated distance can be hard to understand.

In similar fashion to the works cited in literature review of Section 5.2, we fit model property sets for specific scenarios and do not extrapolate to further, untested scenarios. We would like to see how accurate the resulting deformations are in untested scenarios to see if the model properties produce accurate material responses under different external loads.

In Chapter 6, the idea of switching PBDMs at runtime with a view to improving the accuracy of the deformation is presented. This requires, as a pre-processing
5.6 Summary

In this Chapter we provided a procedure to find an optimal set of model properties for any PBDM through the matching of the deformation history of that PBDM with the deformation history of a reference PBDM. We demonstrated the procedure in 3D cantilever and moving cloth scenarios, both of which are popular scenarios in the CG literature. We obtain optimum model property sets for two popular PBDMs: MSS and PBD models each updated in time using first and second order accurate numerical integration schemes. In Chapter 3 we standardised the QVDH procedure, which produces a score \( q \in [0, 1] \) that is 1 when there is a perfect agreement between reference and comparison deformation histories. A primary use of the QVDH procedure is the guided adjustments of the discretisation resolution or model properties of a test PBDM until a satisfactory amount error is achieved. These adjustments can be made manually by the user through trial-and-improvement. However, this is a tedious task, especially when multiple model properties are adjusted over a wide range of potential values. The optimisation method presented here instead automates a procedure to find the best-fit model properties by formulating a minimisation problem with the objective function defined as \( e = 1 - q \in [0, 1] \), where \( q \in [0, 1] \) is the ISO18571 error metric rating used in Chapter 3.

For a tested MSS in the 3D cantilever scenario, the optimised material properties are in pleasing agreement with an \textit{a priori} known analytic solution, showing the correctness of the optimisation method. PBD has no analytic material property equivalence in relation to the FEM material properties \( E, \nu \) as MSS does in
the simulated scenarios. Nevertheless, in the evaluated 3D cantilever and cloth scenarios, the optimisation procedure converges on material property values that produce a rating categorised as “excellent” by the ISO18571 standard metric when suitably good PBD constraints are used. We have shown that the optimisation procedure is an effective one at matching deformation histories of test and reference PBDMs by optimising the model properties of the test PBDM.
Chapter 6

Switching Deformable Models to Improve Accuracy

In Chapters 3 and 4, the quantitative validation of deformation histories (QVDH) software framework was presented. The QVDH framework is a component-based software framework that facilitates the construction of a single scalar quantity, $q \in [0, 1]$, that quantifies the level of agreement between a deformation history of a reference PBDM and a deformation history of a test PBDM. This is a measure of accuracy of the produced deformation of the test PBDM in the selected scenario and over the prescribed duration. In Section 3.3, scores of accuracy of different mass-spring system (MSS) models were calculated in the QVDH framework for a 3D cantilever beam scenario. In Section 5.4, the optimisation of the model properties of both MSS and position-based dynamics (PBD) models was performed by comparing the deformation outputs of these models against the deformation output of a reference finite element method (FEM) simulation solving nonlinear solid mechanics equations. This is achieved by minimising the resulting error, defined as $e = 1 - q \in [0, 1]$, that represents the error of the deformation history of the PBDM being tested. The errors of each of the PBDMs that we evaluated
in the 3D cantilever scenario are presented in Table 6.1.

<table>
<thead>
<tr>
<th>PBDM</th>
<th>Error (3D Cantilever)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFEM</td>
<td>0.109</td>
</tr>
<tr>
<td>CoRotLFEM</td>
<td>0.004</td>
</tr>
<tr>
<td>MSS</td>
<td>0.008</td>
</tr>
<tr>
<td>PBD</td>
<td>0.010</td>
</tr>
<tr>
<td>XPBD</td>
<td>0.038</td>
</tr>
</tbody>
</table>

Table 6.1: We present a summary of the accuracy ratings for the various PBDMs that we tested against the NLFEM reference deformation. We have modelled 3D cantilever beam scenarios in Sections 3.3 and 5.4. The error metric used is the ISO18571 error metric.

This Chapter addresses the idea of improving a test PBDM by switching between more and less accurate PBDMs at runtime. In this way, the PBDM is said to be *adaptive*. Adaptive deformable models change the underlying mathematics, data structures or algorithm, at runtime, in an attempt to comply with some predefined constraints, which depend on the target application. For example, a constraint might be in place to maintain a level of computational performance, or a constraint might be in place to improve accuracy in regions undergoing large nonlinear deformations. This work focuses on combining PBDMs with a view to improving accuracy, as quantitatively validated in the software framework. In particular, the topic of *mixing models* is addressed; that is, when some set of deformable models is used to simulate the deformation of a solid body until some time $T$ and a different set of deformable models proceeds with the simulation for time $t > T$. If a single deformable model is switched out for another single deformable model, it is a *homogeneous* mixed model. Otherwise it is a heterogeneous mixed model. For example, if a MSS is switched for a FEM model, the mixed model is homogeneous. If the MSS is replaced in some regions with an FEM model (so both the MSS and FEM are concurrently modelling different areas of the same body), the mixed model is heterogeneous or a hybrid model.
This work focuses on homogeneous mixed models, whilst heterogeneous mixed models are left as future work.

The goal of mixing PBDMs with a view to improving accuracy comes from the system constraint that requires that the simulation never drops below an error threshold $e_{\text{max}} \in [0, 1]$ as evaluated in the QVDH framework. If the measured accuracy is below this threshold, a more accurate PBDM will be substituted in. Alternatively, it might be the case that the accuracy is unnecessarily large. In this case a less accurate PBDM can (optionally) be used instead if, for example, it is computationally cheaper to use. The active PBDM will either be the gold standard PBDM (which is also used as the reference PBDM in the QVDH framework for comparing against) or a surrogate PBDM that aims to approximate the behaviour of the gold standard PBDM. Surrogate PBDMs are sorted by their accuracy as in Table 6.1 and, therefore, a more accurate model is always available to use, be it another surrogate PBDM or the reference PBDM.

The system constraint could instead (or also) be to maintain a level of computational performance. The different PBDMs have different times to solution, making some more computationally efficient [2][40]. Adapting models for improving computational performance is left as future work. Nevertheless, whichever system constraints might in place, the methodology for switching between models will always be the same: if the predefined constraint is invalidated, the active PBDM will be switched to a more suitable PBDM when possible. The determination of what “when possible” means is characterised in Section 6.2.

The methodology for switching based on accuracy comprises identifying some suitable switching mechanism that would indicate that a different PBDM either

1. should be switched to in order to meet a stricter accuracy demand, or

2. can be switched to whilst still meeting a more lenient accuracy demand.
In this way, the means of indication for switching models may be said to be performed \textit{a priori}: the deformation is potentially going to become increasingly complex and subsequently might require a more accurate model in order to maintain the desired level of accuracy. If the pre-emptive switch is too soon and the indicated point of switching is a false alarm, the system will in turn switch according to (2) above.

6.1 Background

Manteaux \textit{et al}. created a taxonomy of techniques for adapting physically based deformable models in computer graphics, resulting in five categories [96]. These are:

- temporal adaptivity;
- geometric adaptivity;
- basis refinement;
- moving grids and
- mixed models.

Of these five, temporal adaptivity and geometric adaptivity are the most widely used techniques in the literature. The technique being introduced here is method switching and falls into the mixed models category.

Temporal adaptivity entails adapting the time stepping procedure of the simulation. This includes changing the time step value and changing the numerical integration method used to advance the simulation. The problems lie in the stability of the numerical method or in potentially losing detail when a simulation becomes increasingly complex; too large a time step might cause interpenetration
of the body. The CFL condition provides an upper bound of the time step based on the resolution of the mesh to ensure stability in explicit numerical methods (see Section B.2). Implicit numerical methods can be unconditionally stable but can also dissipate energy and introduce artificial numerical damping. Changing the time step can void the energy conservation property of symplectic numerical methods, a problem which has been addressed to allow for multiple time steps across the same mesh [96]. Adapting the time step is an \textit{a posteriori} procedure; the system is rolled back (at the current point in time of the runtime) to a previous state either globally or just locally in the area of interest. This can be a computationally costly procedure, bringing into question the feasibility of the adaptation for real time applications in order to gain some accuracy.

Geometric adaptivity, known also as \textit{h}-refinement, concerns changing the number of nodes used to discretise the body. Using \textit{a posteriori} error estimates such as the $Z^2$ error estimators [41], the resolution can be increased until sufficient accuracy is achieved. Sufficient accuracy is dictated by the convergence of the numerical solution upon an increase in resolution. However, a higher resolution means more degrees of freedom in the system and so a more costly simulation. Moreover, the choice of location for additional nodes can be problematic on unstructured meshes [96].

Basis refinement, known also as \textit{p}-refinement, entails changing the order of the polynomials used to represent an FEM solution in order to improve accuracy. This increases the complexity of the shape functions used to represent the solution and requires more computations but does not require any changes to the mesh. More computations of an FEM are at the expense of computational performance, however, and might be detrimental to real-time computational performance.

Moving grids are used in fluid modelling on a Cartesian grid to increase the local accuracy at the areas of interest. However, they are not particularly suitable
in interactive scenarios in which multiple regions of interest can pop up at once in multiple locations [96].

Mixed models seek to leverage the best aspects of each model whilst ideally avoiding their disadvantageous aspects. However, common strategies for doing this are not easy to determine [96]. Combined PBDMs have been used to simulate the effects of cutting in real-time [20], whilst hybrid models that utilise multiple aspects of different PBDMs have been used with success to better conserve mass [22] and correctly model changes in volume [97].

The strategy presented in this thesis uses some measure across the entire continuum that becomes the input into a switching mechanism that prompts the change of a PBDM. Detail in a deformation can be lost when the PBDM is not sufficiently accurate enough to model complex, nonlinear phenomena. To identify areas of extreme deformation, mesh refinement strategies using the deformation gradient have previously been used [98][99]. The strategy of using the deformation gradient is the strategy that is used in Section 6.2 to trigger a potential active PBDM switch.

### 6.2 Switching mechanism

This Section highlights criteria for switching PBDMs and presents candidate switching mechanisms to facilitate PBDM switching. A suitable switching mechanism is one that ideally

- measures an increasing complex (nonlinear) deformation;
- is a scalar quantity;
- is relatively computationally cheap to compute.
The mechanism should measure the nonlinearities in the deformation of a body effectively to distinguish between more and less complex deformations. If the mechanism produces a scalar quantity then it is easy to set some threshold that triggers a PBDM switch. The mechanism should not be too computationally expensive to compute at each simulation update step because this might hinder real-time performance. If such a mechanism is used then the PBDMs are ready to be mixed. This involves smoothly starting a PBDM from the positional state that the previous PBDM produced, which is discussed next.

6.2.1 Method data exchange

All PBDMs operate on discrete nodal points that discretise some physical body that has some undeformed state, which are the parameterised Lagrangian positions. The deformed state is the collection of (Eulerian) positions at any time of the simulation. As well as nodal positional data, the velocity and acceleration values of the nodes are calculated at each update step of a PBDM. As mentioned in Section 4.2.6, it is common for a PBDM to require some concept of connectivity between nodes of a mesh. For example, nodes located in the local neighbourhood of another node might have some edge connecting them that serves as a constraint in the physical simulation, such as in a MSS or PBD. Alternatively, many nodes might be grouped together into (finite) elements. The connectivity is visible in Figure 6.1 in the form of tetrahedral elements discretising a parallelepiped, with each edge of each tetrahedron connecting two nodes. If the same mesh, together with all of its nodes' associations with one-another, is maintained throughout the simulation, then this nodal data can be input to each and every PBDM throughout the simulation. The “worst-case” scenario for maintaining data relating to node connectivity might be when the FEM requires information of which nodes are
CHAPTER 6. SWITCHING PBDMS

Figure 6.1: A 3D cantilever beam is tetrahedralised by TetGen [47].

contained within each element, which is information that is not needed in a MSS. This information is retained in our system nevertheless to cater for the PBDMs that require more mesh information. Moreover, nodal positional history data is retained for numerical integration schemes of order accuracy two that might require previous positional data as well as current data (for example the BDF2 numerical integration scheme - see Section 2.6). It is possible to extend the system as required.

Each PBDM used for simulation uses the same mesh representation and all undeformed (Lagrangian) nodal positions are the same for all of the PBDMs used. When a PBDM is to be switched to another PBDM, the undeformed, current and previous positions are passed to the new model, as well as the nodal velocities and accelerations. The displacements can be calculated as the difference between deformed and undeformed positions. The time step used between PBDMs also has to be taken into account. Using the same time step between models avoids this problem, however this can lead to poorer computational performance in the cases where a larger time step could be afforded or issues of stability when a smaller time step is necessary [2]. Unless otherwise stated, the numerical integration
scheme used is an implicit trapezoidal method (see Section B.2) and the time step is fixed at $\frac{1}{60}$ seconds, which is the typical time step for real-time applications (see Section 2.3.1).

### 6.2.2 Characterising points of switching: 3D cantilever

The QVDH procedure of Chapter 3 quantifies the accuracy of any deformation due to any PBDM over time. A surrogate PBDM might be switched out for a more accurate PBDM with the goal of increasing the accuracy of the deformation. We require that the mechanism used to switch PBDMs identifies that the deformation is becoming increasingly complex. To characterise suitable switching mechanisms, we use the same 3D cantilever example of Chapters 3 and 5. The 3D cantilever deforms under gravity from rest. The simulation is run for 5 seconds and the material properties are $E = 3500\text{Nm}^{-2}, \nu = 0.25$. The beam oscillates with undamped motion. The QVDH framework diagram is in Figure 6.2. The nonlinear complexity of a deformation is contained within the significant nonlinear

![Figure 6.2: The framework diagram shows a 3D cantilever scenario modelled using NLFEM and LFEM.](image)
displacement gradient terms in the Green-Lagrange strain tensor (see Section 2.4.4), which we use in NLFEM to quantify the amount of strain. These nonlinear terms are truncated in linear FEM, which is valid under small displacements and has the side-effect of volume no longer being conserved when this assumption is violated. This is visible in Figure 6.3, in which the time-evolving plots of the (bidirectional) Hausdorff distance measure of total body displacement of each of the NLFEM and LFEM models are given. The cantilevers deform in an oscillatory motion with a period of around 1 second, but the LFEM has a larger amplitude due to the volumetric growth.
6.2. SWITCHING MECHANISM

Figure 6.3: A 3D cantilever model is modelled using a NLFEM model (left, red) and a LFEM model (right, blue). Note that the volume is not conserved by the linear model.

Screenshots: upper left is the rest state, upper right is 0.5s, lower left is 1.0s, which corresponds to the most displaced configuration and the lower right is 1.5s. The error is $e = 0.109$, which is a result of the volumetric growth observable is the lower left image. This growth is plotted in Figure 6.4.
The error metric score is $e = 0.109$ and is due to this volumetric growth. Figure 6.4 shows the volume proportion of both NFELM and LFEM models. The NLFEM model fluctuates around 1 - which means that volume is almost conserved throughout the deformation - whilst the LFEM model grows to almost 1.5 the initial volume size.

![3D cantilever volume proportion variation over time](image)

Figure 6.4: The volume proportion, defined as the current volume divided by the initial volume, of the cantilevers of Figure 6.3. The volume of the linear model grows to almost 1.5 times the initial volume during each oscillation because the assumptions of small displacements - which leads to the linear theory - is violated.

Switching based on volume conservation

We calculate the approximate volume of the entire beam by summing the volume of each individual tetrahedral element. The volume of the $i^{th}$ tetrahedron is given by:

$$V_i = \frac{1}{6}|(\mathbf{x}_1 - \mathbf{x}_4) \cdot ((\mathbf{x}_2 - \mathbf{x}_4) \times (\mathbf{x}_3 - \mathbf{x}_4))|.$$
6.2. SWITCHING MECHANISM

The approximate volume of the beam is then the summed volume of the elemental tetrahedra:

\[ V_{\text{total}} = \sum_{i=1}^{n} V_i. \] (6.1)

The cantilever returns to its rest state approximately every 1.9 seconds into the simulation as seen in Figure 6.3. Figure 6.4 shows that this is also when the volumes of the both NLFEM and LFEM models are approximately equal to the rest volume. We switch between the NLFEM and LFEM at each of these time points (every 1.9 seconds) and plot the resulting deformation in Figure 6.5. The switching NLFEM/LFEM model models the deformation with an error \( e = 0.064 \), which is an improvement over the simulation using purely LFEM that scored \( e = 0.109 \). We have thus shown that there are points at which the active model can successfully be switched for another model such that:

---

**Figure 6.5:** The 3D cantilever from Figure 6.3 is modelled but this time the LFEM model is switched between a NLFEM model and a LFEM model when the model returns to its rest state. This is approximately every 1.9 seconds. The region shaded in grey is when the NLFEM model is used to deform the test entity, whose displacement is plotted by the dashed curve. The NLFEM reference curve is plotted as a solid curve. The resulting error is \( e = 0.064 \).
• the simulation does not become unstable and
• the error of the simulation is reduced.

We now perform a single switch at a point in time where the rest volume of a body is not conserved using the LFEM model. A suitable time is, for example, at 0.5 seconds, when the LFEM volume proportion is 1.14. The NLFEM model models volume conservation and tries to correct the volumetric growth of the LFEM model, which produces further modes of oscillation and visually looks like an accordion before becoming unstable. In other words, attempting to kick-start the NLFEM model simulation from a state that would otherwise be impossible for it to reach causes the solution to diverge, see Figure 6.6. Use of an implicit backward Euler integration scheme (which is of order accuracy one and is energy dissipative) decays the extra modes of oscillation but at the expense of not reaching the initial rest state again during the deformation, giving an error $e = 0.071$. An implicit Euler integration scheme can be used instead of an implicit trapezoidal integration scheme to maintain stability at the expense of accuracy. These two examples of switching indicate three things:

1. PBDMs can be successfully switched between.
2. If maximum accuracy is important, then volume conservation is important when switching between PBDMs.
3. If the volume is not conserved then implicit backward integration schemes (of order 1 or 2) can still be used to stably simulate the deformation because they are energy dissipating.

So far we have recognised that volume conservation is important in order to switch PBDMs and improve accuracy. However, simply switching in order to conserve
3.2. SWITCHING MECHANISM

Figure 6.6: The 3D cantilever from Figure 6.3 is modelled but this time the LFEM model is switched to a NLFEM model at 0.5 seconds when the volume is not conserved. The region shaded is grey is when the NLFEM model is used to deform the test entity, simulated using implicit Euler and implicit trapezoidal methods. The implicit trapezoidal method is not stable and the solution quickly diverges after the point of switching. The implicit Euler method, however, is stable and produces a deformation with error $e = 0.071$.

volume is not enough to improve the accuracy. Next, we consider a switch based on a measure of (local) deformation.

Switching based on the displacement gradient

As mentioned in the related work of Section 6.1, the works [98] and [99] identify areas of large deformation using the deformation gradient (introduced in Section 2.4.4) and perform mesh refinements in those areas. Recall that the deformation gradient tensor can be written in terms of the displacements as:

$$F = I_3 + \nabla_x u = I_3 + \begin{pmatrix} \frac{\partial u_1}{\partial x_1} & \frac{\partial u_1}{\partial x_2} & \frac{\partial u_1}{\partial x_3} \\ \frac{\partial u_2}{\partial x_1} & \frac{\partial u_2}{\partial x_2} & \frac{\partial u_2}{\partial x_3} \\ \frac{\partial u_3}{\partial x_1} & \frac{\partial u_3}{\partial x_2} & \frac{\partial u_3}{\partial x_3} \end{pmatrix}, \quad (6.2)$$
where $I_3$ is the three-dimensional identity matrix and $\nabla_x u$ is the displacement gradient tensor, mapping undeformed positions, $x$, to their displacements, $u$. The terms in this decomposition correspond to rigid body translation and pure deformation, respectively. The LFEM model can handle neither large relative rotations nor large stretches, so we are really only interested in the pure deformation $\nabla_x u$ here.

We calculate the displacement gradient at the centroid of each element. The centroid of a tetrahedron with four vertices with Lagrangian positions $x_1, x_2, x_3, x_4$ is:

$$x_c = \frac{1}{4}(x_1 + x_2 + x_3 + x_4),$$

and so the displacement gradient at $x_c$ can also be averaged as:

$$\nabla_{x_c} u = \frac{1}{4}(\nabla_{x_1} u + \nabla_{x_2} u + \nabla_{x_3} u + \nabla_{x_4} u).$$

The Frobenius norm is used to assess the size of the displacement gradient at the centroid:

$$||\nabla_{x_c} u||_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^n |d_{ij}|^2} = \sqrt{\text{trace}(\nabla_{x_c} u^T \nabla_{x_c} u)},$$

(6.3)

$d_{ij}$ is the component of $\nabla_{x_c} u$ at the $i^{th}$ row and $j^{th}$ column.

We take as a global measure of the displacement gradient of a body the sum of the size of the displacement gradient (using the Frobenius norm) of each tetrahedron multiplied by its volume, all divided by the entire volume:

$$D_{\text{global}} = \frac{\sum_{i=1}^n V_i ||\nabla_{x_c} u^{(i)}||_F}{V_{\text{total}}},$$

(6.4)

where $\nabla_{x_c} u^{(i)}$ is the size of the displacement gradient of the $i^{th}$ tetrahedron as in
(6.3). This is the Gaussian quadrature rule for integrating across the entire body using one integration point (the centroid of the element).

Returning now to the NLFEM and LFEM simulations (without any switching), the plots of $D_{global}$ for each of the models is given in Figure 6.7. The LFEM model has a larger displacement gradient measure, as we might expect because the displacements are larger due to the volumetric growth. Note that the curves resemble those of Figure 6.3, which makes sense because both sets of curves measure displacement. From Figure 6.7, we can choose thresholds at which to switch between NLFEM and LFEM, corresponding to when the deformation is getting increasingly nonlinear in the calculated displacements. We will use this global displacement gradient measure, $D_{ij}$, in the results of Section 6.3.2 when we switch between MSS and PBD models whilst simulating the deformation of a cloth.

![The global displacement gradient is measured for 3D cantilever beams modelled using NLFEM and LFEM models.](image)

Figure 6.7: The global displacement gradient measure of equation (6.4) is measured for the same 3D cantilever setup used in Figure 6.3: NLFEM and LFEM models are used for the entire simulation without any switching.
Switching based on the elastic potential energy

The elastic potential energy was defined and discussed in Section 2.13 for hyperelastic materials and it depends only on the current deformation (gradient tensor). For NLFEM using a St. Venant-Kirchhoff constitutive law with the Green-Lagrange strain measure \( e \) (not to be confused with the error \( e \)), the strain energy of a single tetrahedron is

\[
W_i = \mu ||e||_F^2 + \frac{1}{2} \lambda \text{tr} e^2.
\]  

(6.5)

The total body strain energy is then taken as the sum of strain energies of all of the elements:

\[
W = \sum_i W_i.
\]  

(6.6)

For the LFEM model, the Cauchy strain, \( \varepsilon \), is used in place of the Green-Lagrange strain tensor, \( e \), in equation (6.5). It is worth noting that calculating the strain energy is a simpler computational procedure than calculating the global displacement gradient measure of equation (6.4). As with our previous example measurement, namely the global displacement gradient measure, we select thresholds to indicate points at which to switch PBDMs. The thresholds are chosen heuristically; more concrete methods for identifying thresholds are left to future work.

Summary

In summary, we switch between PBDMs in order to improve accuracy when

1. volume is sufficiently conserved, and

2. a single scalar measure of the deformation exceeds some chosen threshold.

The two chosen measures in this thesis are:
(a) a global displacement gradient measure, defined in equation (6.4), or
(b) the elastic potential energy of the body, defined in equation (6.6).

The combination of 1 and 2a or 1 and 2b form the *switching mechanism* and this
switching mechanism is used in the results of Section 6.3 to test the improvements
made to the accuracy of a PBDM as a consequence of switching models.

### 6.3 Results

In this Section we test the switching of PBDMs based on the switching mechanism
declared at the end of Section 6.2.2. We test the 3D cantilever with LFEM and
PBD models and note that, because the conclusions drawn from Section 6.2.2
were developed for a specialised scenario of a 3D cantilever, we must also test
the methodology in an unrelated scenario. This new, unrelated scenario is the
hanging cloth scenario that we previously modelled in Chapter 5. In this Section
we simulate the cloth using MSS and PBD models. The 3D cantilever exhibits
periodic, nonlinear deformations that is not damped, whereas the cloth scenario
is damped.

#### 6.3.1 3D cantilever

**Switching LFEM and NLFEM using the elastic potential energy**

We start with the 3D cantilever from the previous Section and switch based on
the elastic strain energy stored in the simulated body. Recall that the error of the
LFEM simulation without any switching is $e = 0.109$. The test PBDM switches
to the NLFEM model when the energy exceeds 5000J and is switched (back)
to the LFEM model when the energy is less than 5000J. At this threshold, the
simulation is unstable when using the implicit trapezoidal method because of
the issues concerning volume that were mentioned in the previous Section. We propose two solutions to this problem that ensure stability.

**Stabilising using implicit Euler integration** The first of these is to use an energy dissipative implicit backward integration scheme, for example implicit Euler, which stabilises the simulation but also damps the motion. The resulting deformation is plotted in Figure 6.8 with error 0.075. Whilst this provides a greater measure of accuracy compared to using only the LFEM model, the artificially damped motion is clearly visible for the LFEM/NLFEM switching model. For scenarios that lasts longer than 5 seconds, the motion would be completely damped.

![Switching FEM models using elastic potential energy](image)

**Figure 6.8:** The test PBDM switches between a LFEM model below a total elastic potential energy of 5000J and a NLFEM model above 5000J. The grey shaded area corresponds to the time during which the NLFEM model is used. Switching introduces some error into the NLFEM simulation (due to volume conservation) and so the NLFEM simulation is not stable using an implicit trapezoidal method. The simulation is stabilised by using an (energy dissipating) implicit Euler integration scheme. We note that the stability issues occur when the PBDM used switches from NLFEM to LFEM, and we can see the energy levels fluctuating around 5000J by the graph area changing rapidly between grey and white before stabilising. The resulting error is $e = 0.075$. 
Stabilising using the constraints of position-based dynamics  The second solution to ensure stability is to exploit the unconditional stability of PBD. We do this by performing an interim switch to PBD that lasts for 10 update steps before eventually switching to the intended target PBDM. This relies on the optimised model properties of PBD being used, which are given in Table 6.1. Using PBD in this way helps to stabilise the simulation, producing an error of only $e = 0.013$.

Remarks  Both of these solutions demonstrate two key points. Firstly, we can obtain solutions of improved accuracy when switching PBDMs in this way. Whilst we do not claim that any of the techniques presented are the ultimate method for mixing models by PBDM switching, we have shown that the methodology is successful at improving accuracy by PBDM switching. Secondly, our proposed framework facilitates rapid exploration with a view to performing experiments such as those carried out in this Section. We can obtain the accuracy resulting from the PBDM switch using the standard QVDH procedure of Chapter 3, which gives instant, objective feedback about the effectiveness of the switching mechanism and (therefore) the switching methodology as a mixed modelling technique.

6.3.2 Cloth

We now return to the hanging cloth scenario that we introduced in the previous Chapter. The cloth is discretised as a (planar) 2D rectangular lattice of quadrilateral elements with $26 \times 26$ nodes. Qualitatively speaking, we would like the cloth to be stretchy, exhibit out-of-plane (transverse) displacements and be susceptible to bending and folding. Therefore we set the material properties as $E = 7000\text{Nm}^{-2}, \nu = \frac{1}{3}$. In addition to gravity, we model “wind” as another external body force that blow into the rear side of the cloth. We present screen-shots of the cloth scenario in Figure 6.9. We use a MSS as the reference PBDM,
which is numerically integrated forwards in time using an implicit backward Euler integration scheme. The MSS internal force computation is nonlinear and some integration schemes such as implicit midpoint and trapezoidal method are only conditionally stable.

We note that it is important for a reference PBDM to be accurate in order to provide reliable reference data, as we did when we validated NLFEM as being a suitable reference PBDM back in Chapter 3. However, we can still test out our switching mechanism for switching PBDMs by evaluating whether or not the accuracy of the switched PBDM increases, as a result of switching, relative to the chosen reference PBDM.

As we did in Chapter 5, we set the spring stiffness constant values as given in [90], which are derived for square elements:

\[
k_{\text{edge}} = \sum_{e} \frac{5}{16} tE = \sum_{e} 4.375 \text{Nm}^{-1} \\
k_{\text{diag}} = \frac{7}{16} tE = 6.125 \text{Nm}^{-1}, \quad (6.7)
\]

where \(k_{\text{edge}}\) is the stiffness of the springs lying across the edges of each square (summed for edges which have elements on either side, i.e. interior element edges), \(k_{\text{diag}}\) is the stiffness for springs lying across the diagonals of the square elements and the cloth has thickness \(t = 0.2\text{mm}\). We note that the stiffness relations in equations (6.7) are derived for an elastostatic problem undergoing in-plane strain and will not necessarily be valid for the deformation we use here. We use the relations nonetheless to test our PBDM switching methodology.
Switching PBD model properties using the displacement gradient.

In this experiment we focus on switching the model properties of the PBD model. The QVDH framework diagram for this experiment is given in Figure 6.10. Since we are using a PBD model, we can adjust the number of solver iterations, which affects the performance directly. In particular, we can switch between a PBD model property set that focuses on performance (through fewer solver iterations) and another model property set that focuses on improved accuracy (through more solver iterations). We optimise the model properties of a PBD model in a pre-processing step (using the optimisation procedure as described in Section 5.3) and present the following model property sets: The set $P_{\text{acc}}$ is for the optimised set of model properties with the smallest error, whilst the set $P_{\text{perf}}$ is the most accurate model property set the optimisation search strategy found with $\leq 10$ iterations of the solver. $P_{\text{perf}}$ thus corresponds to a set that is geared towards improved computational performance, in comparison to $P_{\text{acc}}$. We choose the global displacement gradient threshold (as mentioned in Section 6.2.2) to be$^1$ 0.7; when the global displacement gradient exceeds this value, the tested PBD model uses the $P_{\text{acc}}$ model property set and uses $P_{\text{perf}}$ otherwise. The resulting cloth deformations are illustrated in Figure 6.11. The switched PBDM has an error of 0.082, which is an improvement over the error of 0.104 for the $P_{\text{perf}}$ model property set.

$^1$As mentioned, this threshold is chosen heuristically. We leave more sophisticated threshold selection strategies to future work.

<table>
<thead>
<tr>
<th>Model property set</th>
<th>{iteration count, $k_e$, $k_d$}</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{\text{acc}}$</td>
<td>{20,0.49,0.65}</td>
<td>0.011</td>
</tr>
<tr>
<td>$P_{\text{perf}}$</td>
<td>{7,0.95,0.1}</td>
<td>0.104</td>
</tr>
</tbody>
</table>

Table 6.2: Two model property sets for PBD that favour performance ($P_{\text{perf}}$) and accuracy ($P_{\text{acc}}$).
The cloth scenario has body forces from gravity acting downwards and also wind which is directed towards the camera. The cloth exhibits a lot of bending. The orange cloth on the left is modelled by a MSS model. The blue on the right cloth is modelled by a PBD model. Top image: 0.1s, centre image: 1.0s, bottom image: 1.8s.
Figure 6.10: The framework diagram for a hanging cloth scenario with wind.
Figure 6.11: The reference MSS signal (solid line) is plotted along with the various tested PBD signals. PBD1 is the result of using the $P_{\text{acc}}$ model property set and has an error of 0.011. It is the signal illustrated by long dashes and mostly matches the reference MSS signal. PBD2 is the result of using the $P_{\text{perf}}$ model property set and has an error of 0.109. It is the signal illustrated by short dashes. PBD3 is the result of switching the model property sets depending on if the global displacement gradient measure used passes a chosen threshold of 0.7. When the measured global displacement gradient is greater than 0.7, $P_{\text{acc}}$ is used. Otherwise $P_{\text{perf}}$ is used. PBD3 has an error of 0.082. It is illustrated using alternating long and short dashes. We obtain a closer fit to the reference curve after switching the less accurate model set to the more accurate model set. Below, we have tried to make the distinction between each signal clearer. Finally, the grey area illustrates when PBD3 uses $P_{\text{acc}}$. Otherwise PBD3 uses $P_{\text{perf}}$. 
6.4 Discussion

Modelling (near-)incompressible materials  Different PBDMs conserve volume differently under applied external loads. When switching PBDMs, it is important to switch between PBDMs at a state where the volume is one that is obtainable from the target PBDM. In practice this is typically not something that is known and so it is instead safer to switch when the volume is mostly conserved from the undeformed configuration.

Stabilising at the point of switching  In IVEs, the deformations are particularly challenging due to the fast-twitch interactions that the user gives. Every frame can exhibit multiple, constant external surface loads, for example as a result of externally user-controlled entities in the IVE. Nevertheless, implicit backward Euler is good at stabilising the simulation at the point of switching. We leveraged its energy dissipating features to damp out some of this excess movement before kick-starting the target PBDM. A further avenue to explore is the work of Dinev, Liu and Kavan [100], in which they introduce an integration method that introduces less artificial damping while remaining stable. Alternatively we can use the constraints of PBD to stabilise the simulation by correcting any displacements that are erroneous according to the position-based constraints.

Problems with stabilising  For cloth in particular, we experience stability issues with some integration schemes that are not unconditionally stable. This is due to the cloth bending (that is, the transverse displacements in the cloth simulation). Therefore we are limited to switch between PBDMs that are unconditionally stable, i.e. PBDMs integrated using the methods outlined in [71] and [100] as well as PBD models. These referenced works were researched in order to obtain unconditionally stable simulations of deformations in IVEs.
In addition, switching to a different PBDM for cloth can cause either

- unnatural movements in the resulting deformation. This is alleviated using energy dissipating, backward numerical integration methods such as implicit Euler and BDF2;

- unnatural folds to be removed, which is visible in Figure 6.12.

**Trading performance for accuracy** We have already encountered an opportunity to see the trade-off between computational performance and accuracy. In switching the PBD model in the cloth scenario, we have two choices of model property sets, which were given in Table 6.2. If we choose $P_{\text{perf}}$ we get improved computational performance because we perform fewer constraint solver iterations per update step at the expense of quantitatively validated accuracy. If we instead choose $P_{\text{acc}}$ we get higher accuracy at the expense of more solver iterations and therefore reduced computational performance. We show this in Figure 6.12, in which the two model property sets are switched between in order to improve accuracy. Not only does the cloth more visually similar to the reference after the switch, but we can now quantitatively validate it using the QVDH procedure of Chapter 3. We anticipate similar effects when switching between integration schemes of differing order accuracy using the same PBDM.

### 6.4.1 Limitations and future work

The less accurate PBDMs are used only when the deformation is not very complex. Visually, we can see in Figures 6.8 and 6.11 that the majority of the simulation is performed using the more accurate PBDM. We expect an increase in accuracy over the less accurate PBDM if the majority of the simulation of the switched PBDM is performed using a more accurate PBDM. However, we have chosen
simulations of complex, continuous deformations; in a typical IVE, deformable objects might not always be deforming. We show only that switching is possible using this technique and note that computational performance typically decreases.

We do not measure the performance overheads at the point of switching, which are more noticeable when switching to a more accurate PBDM that it less computationally efficient. For use in real-time, this is a topic that must be addressed. We have instead focused on exploring how accuracy might be improved during a simulation when using a less accurate PBDM. We leave the performance versus accuracy trade-off to future work.

At the start of the Chapter we defined homogeneous and heterogeneous mixed models. A homogeneous mixed model is when we switch a single PBDM for another PBDM such that the nodal displacements are calculated entirely using one PBDM. Otherwise we have a heterogeneous mixed model. In this Chapter we have explored only homogeneous mixed models and so we leave heterogeneous mixed models to future work. We anticipate that heterogeneous mixed models will be an effective mixed modelling technique when we use local deformation measures such as the displacement gradient measure used in Section 6.2.2. This measure quantifies the complexity of the pure deformation in the vicinity of each nodal point and, if we switch PBDMs in areas of differing complexity, then we obtain a hybrid PBDM. This means that multiple PBDMs would work together to determine the unknown nodal displacement field at every update step.

Finally, we have heuristically chosen the switching thresholds when using the global displacement gradient measure in Section 6.2.2 and the elastic strain energy to show that PBDM switching works within our framework. More sophisticated methods for determining thresholds or indeed better switching mechanisms entirely are left to future work.
Figure 6.12: We model the cloth using MSS (left, orange) and PBD (right, blue) models. In the top image, the PBD model properties are $P_{\text{perf}}$, which favours performance via fewer PBD constraint iterations. Note the wrinkle at the bottom of the cloth. In the bottom image, the model property set has been switched to $P_{\text{acc}}$, which is more accurate than $P_{\text{perf}}$. In the case of the PBD material properties here, “more accurate” corresponds to more solver iterations. Within 10 updates the wrinkle has disappeared.
6.5 Summary

In this Chapter we have shown how we can address the topic of mixed modelling using the QVDH procedure and the software framework presented in this thesis. The software framework facilitates the rapid exploration of suitable switching mechanisms to trigger when a switch should happen and which PBDMs should be switched to.

A switch of a PBDM requires only the switching of the test PBDM component. The software framework requires an additional `SwitchingMechanism` component to instigate a PBDM switch. This is a simple extension to the framework presented in Chapter 5. When switching between PBDMs, we should ensure that the model property sets for each PBDM are optimised to a reference PBDM so that the PBDMs (attempt to) model the same material response. We therefore perform a pre-processing optimisation step using the optimisation method in Chapter 5. This can limit our selection of PBDMs to only the optimised PBDMs when we switch models at runtime.

Since we have not addressed performance or memory in these experiments so we can only make conclusions about the accuracy. We have objectively validated the accuracy improvements made using the usual QVDH procedure of Chapter 3 to show that PBDMs can be switched between with a view to improving the resulting accuracy of the deformation. In the 3D cantilever example, we improved the accuracy of the deformation modelled initially by a LFEM model by switching to a NLFEM when the strain energy of the body exceeded a (heuristically) chosen threshold. In addition, we improved the accuracy of a cloth simulation using PBD by switching to a PBD model property set that favours accuracy when the deformation is deemed more complex by using a global displacement gradient measure. When the deformation is less complex we switch to a different PBD
model property set that favours performance. In this way, we begin to see how performance can be objectively traded for accuracy.
Chapter 7

Conclusion

The work in this thesis was motivated by the lack of a standard methodology for evaluating accuracy of PBDMs in the real-time CG literature. PBDMs approximate, to varying degrees, the solutions of the equations of motion of deformable objects from continuum mechanics. FEM is used extensively in engineering disciplines for which real-time simulation is not important but accuracy is. CG is a relatively young research field compared to more mature engineering fields and many new PBDMs are being developed because demand for real-time, interactive deformation simulations is increasing and FEM is not computationally efficient enough to meet the demands. However, in developing more computationally efficient PBDMs, we have noticed a pattern in the accuracy evaluation (or lack thereof) of proposed PBDMs by their authors. We observed that PBDMs were validated as being accurate through two ways. Firstly, PBDMs were qualitatively validated over time, most commonly through visual plausibility, which is necessarily user subjective. Secondly, PBDMs were quantitatively validated but only at a single instance in time. This is an objective measure of accuracy but it does not take into account the deformation history up until the point of evaluation.

In this thesis we have introduced the QVDH procedure that quantitatively
validates any deformation history of any PBDM to obtain a single scalar rating of accuracy (as a percentage) against a reference data set. In Chapter 3 we showed how the QVDH procedure can be used to obtain ratings of accuracy in a 3D cantilever scenario for two MSS models against a reference FEM model. The first of these MSS models has an analytic solution that equates its model properties to those of the FEM model. The QVDH procedure rates the accuracy of this MSS model to 98.5%, which shows the correctness of the procedure. In addition, we can directly compare the accuracy of each PBDM evaluated using the QVDH procedure.

We perform the QVDH procedure inside a component-based software framework, described in Chapters 3 and 4. The components of the framework interact at runtime to allow any PBDM to be evaluated. The framework is conceptually simple, easy to reconstruct and general enough to meet user demands through its replaceable components. However, the performance of the QVDH procedure is bounded by the performance of the PBDMs; if either of the PBDMs do not have real-time performance, then neither will the QVDH procedure because it relies on the data of the PBDMs that is produced at runtime.

In Chapter 5 we present an optimisation procedure to fit model properties of any PBDM such that the resulting deformation agrees with a reference deformation as best as possible. “As best as possible” is quantified by the QVDH procedure. The material response is determined by the model properties of the PBDM and we find that the optimisation procedure finds optimal results in agreement with \emph{a priori} known analytic results, showing that the procedure is effective. However, we find that for complex deformations, the effectiveness of the optimisation procedure depends on the components of the software framework that measure the deformation at each simulation update step. At any state of the deformation, we find that trying to quantify the deformation using a single scalar value can
be a challenge. In particular, by using the Hausdorff distance to measure mesh displacement, a complex deformation will be measured by which nodal point has displaced the most. This has the side effect of other interesting deformation detail being suppressed because the larger nodal displacement dominates. We note, however, that this is a problem with other works in the literature that quantitatively validate PBDMs at a single time instance. We want to measure deformation effectively and with great detail but also we want the quantity to be simple i.e. a single scalar quantity. This is one reason why qualitative (sensory) validation is useful - particularly from subject matter experts. As users, we are good at processing global deformation detail on the body as a whole (which is subjective, qualitative validation) but precise, local deformation detail (as is objectively measured by quantitative validation techniques) is lost on us. In this thesis we have attempted to bridge the gap between the two validation techniques by using the ISO18571 error metric from vehicle safety fields, whose error measures are qualitatively categorised by subject matter experts. As future work we would like to research more appropriate measures of global deformation that better measure the deformation of a whole body at a single point in time whilst being a simple data structure such as a scalar, floating point quantity.

In Chapter 6 we show how the QVDH procedure might be used with adaptive PBDMs by mixing models at runtime. We find that the real-time CG literature does not address the mixing of models with a view to increasing accuracy because there has been no QVDH procedure to validate any decision to switch between models. We successfully switch PBDMs at runtime and increase the accuracy of the resulting deformation, which we can quantify using the QVDH procedure. We introduced two switching mechanisms that are used to trigger a PBDM switch. Whilst the switching mechanisms introduced are not claimed to be the best possible mechanisms, we have shown that it is possible to switch PBDMs at
runtime and increase the accuracy of the deformation. We note that whilst we successfully replace the entire PBDM at each switch, we could instead switch PBDMs in areas of the body undergoing more (or less) degrees of deformation and have multiple PBDMs acting on the discretised body at once. We believe this to be a promising direction in which to take the work presented in this thesis and the QVDH procedure will quantitatively validate the resulting deformation as being more accurate. We leave this to future work.

We hope that the work in this thesis can help to objectively quantify the trade-off between real-time deformable models and their accuracy, whilst also assisting with future research into developing efficient and accurate PBDMs for use in interactive real-time computer graphics applications.
Appendix A

Stiffness Matrices

In this Chapter we derive the stiffness matrices for PBDMs with linearised equations of motion. The equations of motion from equation (2.5) are:

\[ M\ddot{u}(t) + f_{\text{int}}(u(t), t) = f_{\text{ext}}(t). \]

The stiffness matrix is calculated as

\[ K = \frac{\partial f_{\text{int}}}{\partial u} \]

and represents the linear approximation (in \(||u|||\)) of how much changes in displacements of all discretised material points in the body, \(u\), affect the internal forces at each discretised material point in the body, \(f_{\text{int}}\).

For PBDMs with modelling assumptions that make the equations linear in the
nodal displacements, we can write the internal force computation as

\[
\begin{bmatrix}
  f_1 \\
  f_2 \\
  \vdots \\
  f_n
\end{bmatrix}
= K \begin{bmatrix}
  u_1 \\
  u_2 \\
  \vdots \\
  u_n
\end{bmatrix}
= \begin{bmatrix}
  u_1 \\
  u_2 \\
  \vdots \\
  u_n
\end{bmatrix}
\].

(A.1)

where K is the stiffness matrix of the system. The stiffness matrix is symmetric positive-definite and can be decomposed as follows (as explained in [8]):

\[
K = A^TCA,
\]

(A.2)

and so the framework for connecting \( f_{\text{int}} \) to \( u \) through K comprises three distinct steps:

- **Kinematic equation:** \( e = Au, \quad A \in \mathbb{R}^{m \times n}, \)
- **Constitutive law:** \( w = Ce = CAu, \quad C \in \mathbb{R}^{m \times m}, \)
- **Static equation:** \( f_{\text{int}} = A^T w = A^TCAu = Ku, \quad A^T \in \mathbb{R}^{n \times m}. \) (A.3)

These three equations link displacements first to (internal) displacement gradients, then to internal stresses and finally to internal forces at each node of the discretised mesh. This formulation is then used in equation (2.5). In the following Sections we show how this framework is applied to LFEM and 1D MSS models.
A.1 Stiffness matrix assembly for LFEM

The stiffness matrix decomposition $K = A^T C A$ is as follows for LFEM. Each finite element has a stiffness matrix:

$$K_e = \int_V A^T C A dV,$$

to relate the nodal displacements to the internal forces inside each element.

**Kinematic Equation**  The relation $\mathbf{e} = A \mathbf{u}$ links the displacements to the displacement gradients at each node. The Cauchy strain tensor (2.23) takes the following matrix form in three dimensions:

$$\varepsilon = \begin{pmatrix}
    u_{1,1} & \frac{1}{2}(u_{1,2} + u_{2,1}) & \frac{1}{2}(u_{1,3} + u_{3,1}) \\
    \frac{1}{2}(u_{2,1} + u_{1,2}) & u_{2,2} & \frac{1}{2}(u_{2,3} + u_{3,2}) \\
    \frac{1}{2}(u_{3,1} + u_{1,3}) & \frac{1}{2}(u_{3,2} + u_{2,3}) & u_{3,3}
\end{pmatrix},$$

which can be written in Voigt notation as a column vector:

$$\mathbf{e} = [u_{1,1}, u_{2,2}, u_{3,3}, u_{1,2} + u_{2,1}, u_{2,3} + u_{3,2}, u_{1,3} + u_{3,1}]^T,$$

noting that the off-diagonal terms are symmetric and summed. The matrix $A$ has terms that act as the gradient operator by containing shape function derivative terms, and is independent of any (unknown) displacement gradients, so it is constant for each element throughout the simulation. For example, if a body is split into elements each in the shape of tetrahedra containing four nodes with
APPENDIX A. STIFFNESS MATRICES

For linear elasticity, the internal stresses are given by the second Piola-Kirchhoff stress tensor, $\sigma$ and have a linear relationship to the Cauchy strain, $\varepsilon$. This relationship is in general known as a constitutive law and for linear elasticity it is Hooke’s Law, providing the internal stresses in response to the
strains experienced:

\[ \sigma = C : \varepsilon, \]

where \( C : \varepsilon \) is the \textit{direct tensor notation} for multiplying the two tensors \( C \) and \( \varepsilon \). Here, \( C \) is the elasticity tensor. \( C \) is constant and independent of \( u \), thus we can (pre-)compute it prior to the simulation. If the material being modelled is homogeneous (the material response is the same throughout) and isotropic (the response has no preferred direction in which to deform) then the form of \( C \) for tetrahedral elements is:

\[
C \in \mathbb{R}^{6 \times 6} = \frac{E}{(1 + \nu)(1 - 2\nu)} \begin{bmatrix}
1 - \nu & \nu & \nu & 0 & 0 & 0 \\
\nu & 1 - \nu & \nu & 0 & 0 & 0 \\
\nu & \nu & 1 - \nu & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{1}{2} - \nu & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{1}{2} - \nu & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{1}{2} - \nu
\end{bmatrix}.
\]

This uses the following two scalar quantities.

- \( E \) is \textit{Young’s modulus} (or the \textit{elastic modulus}) of the material and describes how stiff the material is (a higher value corresponds to a larger resistance to being deformed elastically) in the direction of the applied load.

- \( \nu \) is \textit{Poisson’s ratio} of the material, which is the (signed) ratio of how much transverse (or contraction) strain is measured compared to axial strain (or longitudinal extension strain in the direction of stretch).

For “physically real” materials, \( E \geq 0, -1 < \nu < \frac{1}{2} \).
Static equation Combining the equation (2.14):

\[ \sigma = \frac{\partial W}{\partial e} \]

and (A.5) gives

\[ W = \varepsilon : C : \varepsilon, \]

which is quadratic in the displacement gradients, \( \nabla_x u \). From this we obtain the\ntangential stiffness matrix:\n
\[ K(u) = \frac{\partial^2 V}{\partial u^2}. \]

This is constant throughout the deformation. This provides a large computational
saving when compared to NLFEM, for which we must compute the tangent
stiffness matrix at every iteration of the Newton-Raphson solver at each update
step.

Similarly to the Cauchy strain, we may write the stress in Voigt notation as:

\[ w = [\sigma_1, \sigma_2, \sigma_3; \sigma_1, \sigma_2, \sigma_3] \]

noting that we have used \( w \) in place of \( \sigma \) to comply with the notation at the start
of this Chapter.

Finally, we can write the quantities of interest as

\[ \text{strain, } e = Au; \]

\[ \text{stress, } w = CAu; \]

\[ \text{internal force, } f_{int} = A^TCAu, \]
A.2. STIFFNESS MATRIX ASSEMBLY FOR MSS

and remark that the matrix $A^T$ acts as the divergence operator in (2.24):

$$f_{\text{int}} = \nabla \cdot \sigma.$$

A.2 Stiffness matrix assembly for MSS

For 3D problems, the computation of $f_{\text{int}}$ in equation (2.26) is nonlinear:

$$f_{\text{int}}^{s_j}(x_i) = -\nabla u_i V = k \frac{u_j - u_i}{||u_j - u_i||} (||u_j - u_i|| - l_s).$$

If the problem is 1D, then it is linear and the internal force computation can be calculated using the $K = A^TCA$ framework since the internal forces are dependent only on the current displacements and act in a straight line and are therefore are linear in $u$.

**Kinematic Equation**  The matrix $A$ in equation (A.2) links the displacements at each node to the spring elongations:

$$e = Au = \delta u.$$

This is the kinematic equation and corresponds to how elongation (or strain) changes with changes in (Lagrangian) coordinates.

**Constitutive Law**  The constitutive law is the physical equation

$$w = Ce = CAu,$$

which links the internal forces of the system, $w$, with the elongations and therefore the displacements using the material properties as in Hooke’s law (2.26). $C \in$
\( \mathbb{R}^{m \times m} \) is a diagonal matrix whose entries are the individual springs’ constant stiffness coefficients.

**Static Equation** The balance equation enforces that in steady state problems of the form \( f_{\text{int}} = Ku = f_{\text{ext}} \) given in (2.7), the internal forces of the MSS must match the external loads placed upon the system in order for equilibrium to be established:

\[
f_{\text{int}} = A^T w = A^T CAu = Ku = f_{\text{ext}}.
\]

The elongations and the displacements are linked through the matrix \( A \), whilst the external forces and the internal forces are linked through \( A^T \).

### A.2.1 Direct stiffness method

The assemblage process of the stiffness matrix concerns the construction of the algebraic equations that describe the displacements achieved under external loads. The stiffness matrix of the total system is assembled as the sum of the stiffness matrices of each element:

\[
K = \sum_{i=1}^{n} K^{(i)}.
\]

Crucially, the stiffness matrix only has non-zero entries for nodes which are joined to another node. Thus for systems with many nodes with few springs connected to each node, the stiffness matrix will be sparse.

**Example** The stiffness matrix for the system depicted in Figure A.1 using the direct stiffness method is to be assembled. There are three masses and two springs, therefore \( K \in \mathbb{R}^{3 \times 3}, A \in \mathbb{R}^{2 \times 3}, C \in \mathbb{R}^{2 \times 2} \). The first spring connects \( m_1 \) and \( m_2 \), which have displacements of length \( u_1 \) and \( u_2 \), respectively. The stiffness matrix for this spring is as follows. The elongation is the length of the difference in
The elements of the above system are springs and the masses $m_1, m_2, m_3$ are subjected to the external loads $f_1, f_2, f_3$. The degrees of freedom of the system are the displacements which are (1-dimensional) scalar quantities. This makes the equations of motion linear, which means that we can solve the system in the $K = A^TCA$ framework.

Displacements: $e_1 = |u_2 - u_1|$. Thus we have:

$$K_1 = \begin{bmatrix} -1 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} k_1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} k_1 & -k_1 & 0 \\ -k_1 & k_1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$  

Similarly for the second spring,

$$K_2 = \begin{bmatrix} k_2 & 0 & -k_2 \\ 0 & 0 & 0 \\ -k_2 & 0 & k_2 \end{bmatrix}.$$  

Finally, the global stiffness matrix is the sum of these element stiffness matrices:

$$K = K_1 + K_2 = \begin{bmatrix} k_1 + k_2 & -k_1 & -k_2 \\ -k_1 & k_1 & 0 \\ -k_2 & 0 & k_2 \end{bmatrix}.$$
Appendix B

Numerical Integration

This Chapter of the appendix supplements the material based on numerical integration techniques that we introduced in Section 2.6.1. We first discuss common finite difference approximations used and then how these approximations are used in the numerical integration schemes that we found to be most common in the CG literature.

B.1 Finite differences

In this Section we present the method of finite differences applied as a difference quotient. In this way, finite differences give a numerical approximation to a continuous derivative.

First differences

For the function \( u(t) \) with first derivative (or slope) \( \frac{du(t)}{dt} \), the derivative is approximated by replacing it with a finite difference over some (time) step size \( h = \Delta x \):

\[
\frac{du}{dt} \approx \frac{\Delta u}{\Delta t}.
\]
The values of $u(t)$, $u(t + h)$ and $u(t - h)$ may be used. Three basic possibilities for $\Delta u/\Delta t$ are:

- **Forward difference**, $\Delta_F = \frac{u(t + h) - u(t)}{h}$;
- **Backward difference**, $\Delta_B = \frac{u(t) - u(t - h)}{h}$;
- **Central difference**, $\Delta_C = \frac{u(t + h) - u(t - h)}{2h}$.

The "one-sided" differences, $\Delta_F$ and $\Delta_B$, are first order accurate, whilst the central difference is second order accurate. The accuracy can be understood by examining the Taylor series of the one-sided differences, which are:

Forward:
$$u(t + h) = u(t) + hu'(t) + \frac{1}{2}h^2u''(t) + \frac{1}{6}h^3u'''(t) + ...; \quad (B.1)$$

Backward:
$$u(t - h) = u(t) - hu'(t) + \frac{1}{2}h^2u''(t) - \frac{1}{6}h^3u'''(t) + ... \quad (B.2)$$

Subtracting $u(t)$ dividing by $h$ in both (B.1) and (B.2) yields (note the rearrangement in the latter):

Forward:
$$\frac{u(t + h) - u(t)}{h} = u'(t) + \frac{1}{2}hu''(t) + \frac{1}{6}h^2u'''(t) + ...; \quad (B.3)$$

Backward:
$$\frac{u(t) - u(t - h)}{h} = u'(t) - \frac{1}{2}hu''(t) + \frac{1}{6}h^2u'''(t) - ... \quad (B.4)$$

The error is in the excess terms on the right hand sides of (B.3) and (B.4). For $|h| < 1$, the leading error terms are $\frac{1}{2}hu''(t)$ and $-\frac{1}{2}hu''(t)$, respectively. The first power of $h$ causes first order accuracy. The leading error term in the central difference is found by subtracting (B.2) from (B.1) and dividing through by $2h$ (since we are working over two finite differences):

Central:
$$\frac{u(t + h) - u(t - h)}{2h} = u'(t) + \frac{1}{6}h^2u'''(t) + ... \quad (B.5)$$
APPENDIX B. NUMERICAL INTEGRATION

The error is in a term with $h^2$, giving second order accuracy. The central difference is more accurate than both one-sided differences, therefore, when $h = \Delta t$ is small.

Second differences

We can numerically approximate second order derivatives using first order finite difference expressions in a recursive manner. For instance, the second-order central difference is obtained by centering over forward and backward first difference approximations:

$$\text{Central: } u''(t) \approx \frac{u(t+h) - u(t-h)}{2h} - \frac{u(t) - u(t-h)}{h} = \frac{u(t+h) - 2u(t) + u(t-h)}{h^2}. \quad (B.6)$$

B.2 Numerical integration schemes

In this Section we provide derivations for the numerical integration schemes summarised in Table 2.2 of Chapter 2.

Explicit Euler method

This method steps forwards into the future, using the displacement at the current time step:

$$Mv_{t+h} = Mv_t + h(f_{\text{ext}} - f_{\text{int}}(u_t))$$

$$u_{t+h} = u_t + hv_t. \quad (B.7)$$

The lumped mass matrix $M$ of equation (2.5) is diagonal and so the inverse $M^{-1}$ both exists and is trivial to compute, resulting in a series of reciprocal entries.
along the diagonal:

\[ \mathbf{v}_{t+h} = \mathbf{v}_t + h \mathbf{M}^{-1} (\mathbf{f}_{\text{ext}} - \mathbf{f}_{\text{int}}(u_t)). \]

Everything on the right hand side is known and so the method is termed explicit. Explicit methods are *conditionally* stable. To see this for the explicit Euler method, we employ the so-called **linear test equation**:

\[ \frac{dy(t)}{dt} = \lambda y(t), \]

with solution

\[ y(t) = e^{\lambda t}. \]

Using the explicit Euler method, this gives:

\[ y_{t+h} = y_t + h \lambda y_t = (1 + h \lambda) y_t, \]

which is stable if, and only if, \(|1 + h \lambda| \leq 1\). The method is first order accurate and so necessitates such a very small time step in order to remain both accurate and stable, that it is not suitable for use in a real-time CG application.

**The Courant-Friedrichs-Lewy (CFL) condition** provides an upper bound of the time step required based on the resolution of the mesh to ensure stability in explicit numerical methods [101].

**Explicit symplectic Euler method**

The explicit Euler method can be made *symplectic* (which means that the error in the solution will neither grow or decay over time i.e. the method is *energy conserving*) [8] by integrating the displacements using the velocity from the next
APPENDIX B. NUMERICAL INTEGRATION

update step:

\[
M v_{t+h} = M v_t + h(f_{\text{ext}} - f_{\text{int}}(u_t))
\]

\[
u_{t+h} = u_t + hv_{t+h}.
\] (B.8)

This is also called a semi-implicit Euler method.

Implicit Euler method

The implicit backward Euler method safely steps from the current solution to the next since the displacements are from at the next time step in (2.27):

\[
M \frac{v_{t+h} - v_t}{h} + f_{\text{int}}(u_{t+h}) = f_{\text{ext}}.
\] (B.9)

The implicit step calculates the velocity as if the next displacement had already been computed; it assumes it implicitly:

\[
v_{t+h} = \frac{u_{t+h} - v_t}{h}.
\]

Rearranging to the forward difference approximation to the (next) displacement (B.8):

\[
u_{t+h} = u_t + hv_{t+h}
\]

and substituting into the above implicitly governed equations (B.9) gives

\[
M \frac{v_{t+h} - v_t}{h} + f_{\text{int}}(u_t + hv_{t+h}) = f_{\text{ext}}.
\] (B.10)
The Taylor expansion of the internal force terms around \( u_t \) is:

\[
\begin{align*}
\mathbf{f}_{\text{int}}(u_t + h \mathbf{v}_{t+h}) &= \mathbf{f}_{\text{int}}(u_t) + \left( \frac{\partial \mathbf{f}_{\text{int}}(u_t)}{\partial u_t} \right) h \mathbf{v}_{t+h} + \frac{h^2}{2} \mathbf{v}_t^T \left( \frac{\partial^2 \mathbf{f}_{\text{int}}(u_t)}{\partial u_t^2} \right) \mathbf{v}_t + O(h^3).
\end{align*}
\]

The linear approximation to this is

\[
\begin{align*}
\mathbf{f}_{\text{int}}(u_t + h \mathbf{v}_{t+h}) &\approx \mathbf{f}_{\text{int}}(u_t) + h \mathbf{K} \mathbf{v}_{t+h},
\end{align*}
\]

and the \textit{Jacobian} is the stiffness matrix, \( \mathbf{K} \). Substituting the Jacobian into (B.10) gives

\[
\begin{align*}
\mathbf{M} \mathbf{v}_{t+h} - \frac{\mathbf{v}_t}{h} + \mathbf{f}_{\text{int}}(u_t) + h \mathbf{K} \mathbf{v}_{t+h} &= \mathbf{f}_{ext},
\end{align*}
\]

which is, upon rearranging:

\[
\begin{align*}
(M + h^2 \mathbf{K}) \mathbf{v}_{t+h} &= \mathbf{M} \mathbf{v}_t + h(\mathbf{f}_{ext} - \mathbf{f}_{\text{int}}(u_t)) \\
\mathbf{u}_{t+h} &= \mathbf{u}_t + h \mathbf{v}_{t+h}.
\end{align*}
\] (B.11)

The method is implicit because the unknown velocity can appear on both sides of the equation. It is first order accurate also, but is unconditionally stable. To see this, consider the implicit Euler solution to the linear test equation:

\[
\begin{align*}
y_{t+h} &= y_t + h \lambda y_{t+h} = \frac{1}{1 - h \lambda} y_t.
\end{align*}
\]

It can be seen that

\[
\left| \frac{1}{1 - h \lambda} \right| \leq 1
\]

for any step size \( h \); no artificial energy is injected into the system (due to the numerical approximations to the derivatives). The implicit Euler method does
experience numerical damping, however, which damps the energy of the system artificially over time [8].

**Implicit trapezoidal method**

With a view to better approximate the internal forces at the next update step, all possible values are averaged between their current and next values:

\[ u = \frac{u_t + u_{t+h}}{2}. \]

Inserting this into (2.27) gives:

\[ Mv_{t+h} - v_t \frac{1}{h} + \frac{1}{2}(f_{\text{int}}(u_t) + f_{\text{int}}(u_{t+h})) = f_{\text{ext}}. \]

As in the implicit backward Euler method, the forward difference is used for \( u_{t+h} = u_t + hv_{t+h} \) in the internal force terms:

\[ \frac{1}{2}(f_{\text{int}}(u_t) + f_{\text{int}}(u_{t+h})) = \frac{1}{2}(f_{\text{int}}(u_t) + f_{\text{int}}(u_t) + hKv_{t+h}) \]

to give the equations:

\[ (M + \frac{h^2}{2}K)v_{t+h} = Mv_t + h(f_{\text{ext}} - f_{\text{int}}(u_t)) \]

\[ u_{t+h} = u_t + hv_{t+h}. \]  \hspace{1cm} (B.12)

The idea of taking the central difference here (from the midpoint of \( f_{\text{int}}(u_t) \) and \( f_{\text{int}}(u_{t+h}) \)) means that the method is second order accurate by differencing with
the fully implicit internal force at the next time step:

\[
\begin{align*}
f_{\text{int}}(u_{t+h}) - f_{\text{int}}(u_t) &= \frac{h}{2}(f_{\text{int}}(u_t) + f_{\text{int}}(u_{t+h})) \\
&= f_{\text{int}}(u_t) + \frac{h}{2}(f_{\text{int}}(u_t) + f_{\text{int}}(u_{t+h})) + O(h^3) \\
&- f_{\text{int}}(u_t) - \frac{h}{2}(f_{\text{int}}(u_t) + f_{\text{int}}(u_{t+h})) = O(h^3).
\end{align*}
\]

Accuracy gains in the truncation error come at the cost of performing more computations in evaluating the new velocities and displacements.
Bibliography


[81] Eder Miguel, Derek Bradley, Bernhard Thomaszewski, Bernd Bickel, Wojciech Matusik, Miguel A Otaduy, and Steve Marschner. “Data-driven


