Cartilage Simulation using Smoothed Particle Hydrodynamics

by

Philip Boyer, B.Eng.

A thesis submitted to the
Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of
Master of Applied Science in Biomedical Engineering

Carleton University
Department of Systems and Computer Engineering
Ottawa, Ontario, Canada

© 2013, Philip Boyer
Abstract

Abnormal bone growth in the hip joint causing increased stress during motion is a condition known as Femoral Acetabular Impingement (FAI). FAI is considered to be a primary cause of osteoarthritis in this joint due to wear of articular cartilage. A computer simulation for preoperative evaluation of FAI requires the representation of cartilage for accurate force and stress determination, but current methodologies such as the finite element method (FEM) do not simultaneously provide both the accuracy and the computational speed necessary for such a representation. In this thesis, a fast and accurate simulation of articular cartilage is proposed using adaptations of previous research and unique extensions to the method of smoothed particle hydrodynamics (SPH). Strong correlation is found between simulations of compression and indentation experiments of cartilage with previously published experimental results, with simulations operating in excess of real-time rates.
Acknowledgements

First of all, I would very much like to thank my supervisor, Professor Chris Joslin, without whose guidance this research would have long ago tumbled into a limbo of confusion and mediocrity. I would also like to thank Professor Andrew Speirs for his assistance in the interpretation of the component stresses of the cartilage simulations. Finally, I would be remiss in not mentioning Sean LeBlanc, whose collaborations in C++ optimization and Maya visualization were instrumental in this research attaining its current level of quality.
Table of Contents

Abstract........................................................................................................................................... i
Acknowledgements ......................................................................................................................... ii
Table of Contents ........................................................................................................................... iii
List of Tables .................................................................................................................................. vi
List of Figures............................................................................................................................... vii
List of Abbreviations ..................................................................................................................... xiii

Chapter 1 Introduction ..................................................................................................................... 1
  1.1 Motivation ............................................................................................................................... 1
  1.2 Problem Statement .................................................................................................................. 2
  1.3 Contributions of this Research ............................................................................................... 4
  1.4 Organization of this Thesis ..................................................................................................... 5

Chapter 2 Background and Literature Review ............................................................................... 7
  2.1 The Hip Joint .......................................................................................................................... 7
    2.1.1 Anatomy of the Hip........................................................................................................... 7
    2.1.2 Articular Cartilage Composition and Behaviour ............................................................... 8
    2.1.3 Femoral Acetabular Impingement .................................................................................... 12
  2.2 Deformable Computer Modeling .......................................................................................... 13
    2.2.1 State of the Art ............................................................................................................... 13
    2.2.2 Smoothed Particle Hydrodynamics ................................................................................ 18
    2.2.3 Cartilage Computational Models ................................................................................... 23

Chapter 3 Methodology .................................................................................................................. 29
  3.1 Integration Methods ................................................................................................................ 30
  3.2 Neighbour Search Algorithms ............................................................................................... 34
### Chapter 3

3.2.1 Parallel Octree Neighbour Search .................................................. 34

3.2.1.1 Particle Sorting .............................................................................. 36

3.2.1.2 Tree Creation ................................................................................ 39

3.2.1.3 Tree Traverse ............................................................................... 40

3.2.2 Cell Indexing .................................................................................... 41

3.3 Elastic Solids in SPH ......................................................................... 44

3.4 Rotation Handling ............................................................................... 47

3.5 Matrix Diagonalization ...................................................................... 51

3.6 Articular Cartilage Property Implementation ..................................... 53

3.6.1 Neo-Hookean Solid Matrix ................................................................. 54

3.6.2 Collagen Fibril Network ................................................................... 55

3.6.3 Porosity-Dependent Fluid Model ..................................................... 60

3.6.4 Fluid Mass Exchange With Cartilage Exterior ................................. 67

3.7 Rigid Body Boundary Enforcement ..................................................... 70

3.8 C++ Program Framework ................................................................. 73

### Chapter 4

Results of Elasticity and Cartilage Compression Experiments ............ 79

4.1 Elasticity Simulations ......................................................................... 79

4.2 Cartilage Simulations .......................................................................... 85

4.2.1 Indentation Experiment A ................................................................. 86

4.2.2 Indentation Experiment B ................................................................. 97

4.2.3 Unconfined Compression Experiment ............................................. 107

4.2.4 Cartilage Experiments Summary .................................................... 117

### Chapter 5

Conclusions and Future Work ................................................................. 120

5.1 Conclusions ......................................................................................... 120

5.2 Future Work ....................................................................................... 122
List of References ....................................................................................................................... 127
Related Publications ..................................................................................................................... 139
List of Tables

Table 3.1 Collagen fibre orientation according to zone composition. Depth is measured from cartilage surface, and fibre angle is measured from a horizontally aligned cartilage surface. ................................................................. 56

Table 4.1 Material parameters and values used in the cartilage compression simulations. ................................................................. 89

Table 4.2 Summary of results for all cartilage compression experiments. ......................... 117
List of Figures

Figure 2.1 Cross-section of the hip joint showing anatomical features (reproduced from [7])................................................................. 8

Figure 2.2 Microscopic composition of articular cartilage. ............................................. 9

Figure 2.3 Mechanical properties of cartilage layers.......................................................... 11

Figure 2.4 Impingement morphologies and contact areas due to osseous growths (red circles): normal condition (left), cam impingement (centre), and pincer impingement (right). ................................................................................................. 13

Figure 2.5 Illustration of how a continuum can be divided into a series of mass-springs to simulate solid object behaviour. ................................................................. 15

Figure 2.6 Example division of material model into finite elements with colourization according to calculated stress. (For illustrative purposes only, not intended to represent an actual FEM discretization). ................................................................. 16

Figure 2.7 Incorrect FEM under large rotational deformations with inflated head (left) and correct deformation on the right (reproduced from [47]). ........................................... 17

Figure 2.8 Smoothed particle hydrodynamics smoothing function with Gaussian support in a 3-dimensional domain. W is the smoothing kernel function. i is the current particle. h is the smoothing length. xi and xj are the positions of the particle and its neighbour, respectively. ......................................................................................... 19

Figure 2.9 Still frame of SPH fluid breaking dam simulation using the procedure implemented in this research, with visualization by S. Leblanc................................. 20
Figure 2.10 Blood flow out of an injured artery. Blood is represented by SPH fluid particles and arterial wall is composed of a tetrahedral mesh (reproduced from [69]).

Figure 2.11 Arcade model of Benninghof. Collagen fibres originate and terminate on subchondral bone and are horizontal in the superficial zone.

Figure 3.1 Butcher tableau for 4th-order Runge Kutta integration method.

Figure 3.2 Example of hierarchical breakdown of nodes from the root in a 1-dimensional representation of a 3-dimensional octree. As levels increase, each of the preceding level’s cells are subdivided into eight more cells, unless a cell does not contain any particles (black) or contains less than a critical number (green). Lower levels cells are the children of the parent cells they are formed from.

Figure 3.3 Example C++ bitmasking operations provided for illustration of the method to extract the 64-bit integer z-index from three-dimensional Cartesian coordinates.

Figure 3.4 Radix sort example. Algorithm follows a top-to-bottom approach from input to output.

Figure 3.5 Two-dimensional representation of volume space subdivided into grid cells for neighbour search algorithm with blue dots representing particles.

Figure 3.6 Visualization of corotated rotation handling of particle neighbourhoods. Without the corotated approach, a rotation of the initial configuration is falsely interpreted as a displacement (centre). Inclusion of the corotated approach allows for rotations without creating additional forces (right).

Figure 3.7 Method for calculation of collagen fibre contribution based on orientation. The horizontal green arrow represents the assumed collagen fibre orientation (superficial
zone). The distance between particle i and neighbour j is represented by the light grey arrow. The stress contribution is adjusted by the ratio represented by the blue line. 57

Figure 3.8 Spring model representing a collagen fibre: a linear spring with stiffness $E_0$ in parallel with a nonlinear spring with stiffness $E_i = E_0 \varepsilon_f$. 59

Figure 3.9 Example fluid mass exchange between SPH particles. Dark blue arrows represent pore pressure, green arrows represent capillary pressure and light blue arrows represent resulting fluid flow. 61

Figure 3.10 Void particle instantiation: The current particle position (blue) and the positions of its neighbouring particles (green) are checked to determine where void particles (white) should be placed. A) Four positions are currently unoccupied by particles, so four void particles are created at those positions. B) Seven positions are currently unoccupied by particles, so seven void particles are created at those positions. 68

Figure 3.11 Boundary enforcement: deformable SPH particles (blue) are prevented from penetrating the smoothing lengths of the boundary particles (green), which are given support by two layers of virtual particles (gray), resulting in a boundary that traces along the smoothing lengths (purple). 71

Figure 3.12 The algorithm used for simulation of deformable solids with the current procedure. 74

Figure 4.1 Solid rod elasticity and rotation test: $t = 0s$ and $t = 0.5s$. Young’s modulus = 10 kPa. 81

Figure 4.2 Falling wedge rotation test: $t = 0s$ and $t = 0.5s$. Young’s modulus = 10 kPa. 82
Figure 4.3 relationship between the number of simulated particles versus frame rate of elasticity simulations.................................................................................................................. 84

Figure 4.4 Reaction force response in indentation experiment A: elastic and biphasic simulations versus reported experimental (1624 particles) ........................................... 90

Figure 4.5 Visualization of stress distribution of indentation experiment A. A) Initial condition (0 s). B) Maximum strain (2.38 s), C) End of relaxation (3.88 s), D) End of simulation (6.26 s). (1624 particles) ................................................................................... 91

Figure 4.6 Fluid mass of a particle in the indented region over the duration of the biphasic simulation of indenter experiment A. ......................................................................................... 93

Figure 4.7 Component stresses of biphasic simulation of indenter experiment A. (1624 particles) .............................................................................................................................. 94

Figure 4.8 Fibril stress distribution of indenter experiment A at 2.38 s: A) front view, B) opposite view. ............................................................................................................................................ 96

Figure 4.9 Pore stress distribution of indenter experiment A at 2.38 s: A) front view, B) opposite view. ........................................................................................................................................... 98

Figure 4.10 Reaction force response in indentation experiment B: elastic and biphasic simulations versus reported experimental. (3206 particles) .................................... 99

Figure 4.11 Reaction force response in indentation experiment B with increased surface area: elastic and biphasic simulations versus reported experimental. (6696 particles) .................................................................................................................................................. 101

Figure 4.12 Reaction force response in indentation experiment A with increased surface area: elastic and biphasic simulations versus reported experimental. (4774 Particles) .................................................................................................................................................. 102
Figure 4.13 Visualization of stress distribution of indentation experiment B. A) Initial condition (0 s), B) Maximum strain (2.38 s), C) End of relaxation (3.88 s), D) End of simulation (6.26 s). (6696 particles) ................................................................. 103

Figure 4.14 Pore stress distribution of indenter experiment B (increased surface area): A) Maximum strain (2.38 s), B) End of simulation (6.26 s), noting the faster recovery in the area opposite the offset indenter particles ................................................................. 105

Figure 4.15 Irregularities of the geometry of the indenter used in indenter experiment B due to method of Maya initialization: A) Particles offset to one side and extra vertical column of particles, B) Missing lower particle and single particle column compared to three columns on opposite side ................................................................. 105

Figure 4.16 Fibril stress distribution of indenter experiment B (increased surface area) at 2.38 s: A) front view, B) opposite view, noting the increased fibril stress in the area of the offset indenter particles (left). ............................................................................. 106

Figure 4.17 Unconfined compression 1%/s strain rate. Comparison of reaction force results for simulations with active biphasic properties without void particles (“No Void”), with void particles (“Void”), and basic elastic versus reported experimental. .......................................................................................................................... 109

Figure 4.18 Unconfined compression with biphasic (void). A) Initial configuration (0 s), B) Maximum strain (15 s), C) End of simulation (30 s) ................................................................. 110

Figure 4.19 Component stresses for 1%/s strain rate unconfined compression. ........... 111

Figure 4.20 Component stresses for 1%/s strain rate unconfined compression with calculated total material stress. ........................................................................................................... 112
Figure 4.21 Component stresses for 15%/s strain rate unconfined compression with calculated total material stress. ................................................................. 113

Figure 4.22 Matrix axial stresses of 15%/s strain rate unconfined compression divided into layers. .............................................................. 114

Figure 4.23 Calculated total material stress versus strain for 15%/s and 1%/s strain rate unconfined compression simulations. ................................................................. 115

Figure 4.24 Component stress distributions of 1%/s strain rate compression experiment at maximum strain: A) Fibril stress, B) Pore stress. ...................................................... 116

Figure 4.25 Comparing elastic versus biphasic results for frame rate versus number of particles for all cartilage indenter simulations. ................................................................. 118
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AABB</td>
<td>Axis-aligned Bounding Boxes</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BEM</td>
<td>Boundary Element Method</td>
</tr>
<tr>
<td>BPVE</td>
<td>Biphasic Poroviscoelastic</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>CT</td>
<td>Computed Tomography</td>
</tr>
<tr>
<td>FAI</td>
<td>Femoral Acetabular Impingement</td>
</tr>
<tr>
<td>FEM</td>
<td>Finite Element Method</td>
</tr>
<tr>
<td>fps</td>
<td>Frames Per Second</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphics Processing Unit</td>
</tr>
<tr>
<td>MEL</td>
<td>Maya Embedded Language</td>
</tr>
<tr>
<td>MRI</td>
<td>Magnetic Resonance Imaging</td>
</tr>
<tr>
<td>NEM</td>
<td>Natural Element Method</td>
</tr>
<tr>
<td>RAM</td>
<td>Random-Access Memory</td>
</tr>
<tr>
<td>RGB</td>
<td>Red/Green/Blue</td>
</tr>
<tr>
<td>SPH</td>
<td>Smoothed Particle Hydrodynamics</td>
</tr>
<tr>
<td>SVD</td>
<td>Singular Value Decomposition</td>
</tr>
<tr>
<td>TMM</td>
<td>Tensor Mass Model</td>
</tr>
</tbody>
</table>
Chapter 1  Introduction

1.1 Motivation

Femoral Acetabular Impingement (FAI) is a medical condition where osseous growths on the femur or the acetabular rim cause supraphysiological stress concentrations in the cartilage on the femoral head and in the interior of the acetabulum. During motion and loading these impinged areas have a higher probability of incurring damage and wear, which is considered to be a primary cause of the degeneration of cartilage leading to osteoarthritis [1][2]. Preoperative identification of an impingement state within the hip joint would provide considerable advantage in cases of total hip arthroplasty where potential points of conflict should be addressed prior to the procedure to elucidate how best to restore normal range of motion to the patient.

An interactive computer simulation of the acetabulum is one tool physicians could potentially use for identification of FAI. Such simulations would ideally be generated based on patient anatomy, allowing for cases to be tailored to individual requirements, and for the effect of potential changes to the anatomical model to be viewed in real-time. Computed Tomography (CT) scans have previously been employed to develop computer simulations of the acetabulum [3][4], however CT does not provide adequate
identification of soft tissue, specifically damage to the articular cartilage that is the primary concern in cases of osteoarthritis. Magnetic Resonance Imaging (MRI) does however provide visualization of soft tissue and could theoretically be instrumental in the development of a superior simulation that would provide as yet unreachable levels of accuracy in terms of reaction forces and stress concentration identification due to FAI in the large scale evaluation of a patient-specific acetabulum simulation.

The development of the proposed simulation is fundamentally dependent on the creation of a model to represent cartilage that allows for the determination of responses to impingements in the form of numerical results that can be updated in a real-time fashion. This remains an open problem. While there are many examples of research that has focused on the accurate evaluation of cartilage in compression experiments, they have been in the past without exception based upon finite element methods (FEM). FEM is known for its inability to resolve large scale simulations in a timely manner, especially when nonlinear soft tissue components are included in the process, and so is mutually exclusive to the design of a preoperative evaluation tool. The task then is to develop a new method for simulating cartilage that could potentially be used in an interactive simulation of FAI that can also provide the desired numerically accurate results. Due to the wide variation in material properties between cartilage samples, it is suggested here that accurate can be defined as being within ten percent of the peak reaction force when applied to a sample with known material parameters, and within an order of magnitude when the material parameters are assumed based on a pre-existing set.

1.2 Problem Statement
The task of developing a new method for the simulation of articular cartilage can be summarized by the following:

a) *Identify an appropriate computer graphics modeling method*

   The method must be capable of representing a soft-solid with behaviour that is firmly grounded in real physics so accurate numerical results to the sub-millimeter level can be retrieved. It must be capable of functioning at a computational speed that greatly exceeds similar FEM simulations. The method must be capable of being adapted to the application of nonlinear material parameters. The method must be capable of performing experiments to validate the procedure.

b) *Extend the method to the simulation of cartilage*

   Components or methods extending the underlying modelling method for the accurate simulation of cartilage must be created or adapted before being included in the simulations.

c) *Optimize the procedure to perform simulations at a high computational speed*

   The cartilage simulation must function at a level commensurate with achieving the eventual goal of applying the model to an interactive simulation of the acetabulum. Practices and methods for use in the design must be evaluated for their computational efficiency before being implemented.

d) *Obtain and validate numerical results from cartilage simulations*

   The procedure must be capable of being validated against published experimental results by providing numerically accurate responses based on real physical scenarios. High stress regions must be able to be identified and reaction forces to deformations must be computable.
1.3 Contributions of this Research

A variety of contributions are proposed to achieve the requirements for successful resolution of the problem as described in Section 1.2:

a) Extension of an elastic solid Smoothed Particle Hydrodynamics (SPH) simulation to the CUDA parallel programming framework

A C++ program devised to implement the procedure based upon the elastic solid SPH method presented by Becker et al. [5] is developed to operate using the CUDA parallel programming framework, thereby considerably increasing the computational speed of the simulation.

b) Cartilage components extended to SPH

The work of Lenaerts et al. [6] on the representation of porous materials in SPH is significantly adapted to more adequately represent the biphasic (solid/fluid) nature of articular cartilage. The work is extended to apply to completely saturated materials undergoing deformation and equations are reworked to be more explicitly dependent on the force disparity between particle neighbours. Fluid mass is exuded from the cartilage during compression into an infinite external volume referred to here as void particles, and mass is similarly reabsorbed by the cartilage from void particles during swelling. Void particle instantiation does not require explicit fluid SPH particle creation and so significantly reduces the computational burden associated with simulation of a biphasic material. A method is proposed to represent the contribution of the collagen fibril network to the total material stress of cartilage in SPH. Collagen fibres are defined between SPH particle neighbours with orientations that change
depending on the deformation. Depth dependent orientation of fibrils is also incorporated. Individual collagen fibre stress response is determined according to previously published research in FEM.

c) Experiments to validate the procedure for simulation using SPH

Force response to a rigid body causing deformation is calculated through a simple application of Newton’s second law to the forces experienced by particle neighbourhoods within the range of influence of particles comprising the aforementioned rigid bodies. Elastic solid simulations are verified with experiments modeled on the simulations performed by Becker et al. [5] to ensure the material behaves according to that dictated by real-world conditions, including correct resolution of rotations. Unconfined compression and indentation experiments of the proposed cartilage model are performed and their numerical results are validated for accuracy against published research.

1.4 Organization of this Thesis

This thesis is organized as follows: Chapter 2 provides a basic understanding of hip joint anatomy, the composition of cartilage, and FAI, as well as reviews of related research in the fields of deformable computational modeling, SPH, and cartilage computational models. Chapter 3 outlines the methodology employed in adapting the SPH elastic solid model to cartilage, and describes the functionality of the program designed to run the simulations. In Chapter 4, the proposed procedure is evaluated based on results from elasticity and cartilage compression experiments. Chapter 5 establishes conclusions about
the current research and discusses considerations related to possible future directions.

Related publications by the author are mentioned at the end of the thesis.
Chapter 2  Background and Literature Review

Reviewed in brief is a morphology of the hip and the associated articular cartilage, followed by a detailed investigation of previous research in the area of deformable computer modeling as well as a review of simulation methods relating to cartilage.

2.1  The Hip Joint

2.1.1  Anatomy of the Hip

The hip (Figure 2.1) is a ball and socket synovial joint formed by the intersection of the femur (ball) and the os coxa, or hip (socket). Both the femoral head and the acetabulum are considered to be near-spherical, sharing opposing surfaces with a common radius of approximately 25 mm and a deviation of sphericity on the order of < 100 µm [8]. Both surfaces have a layer of articular hyaline cartilage, of which the layer on the acetabulum is referred to as the acetabular labrum. A soft tissue known as the synovial membrane between the opposing cartilage layers acts as a container for synovial fluid that has mechanical properties providing lubrication during movement and impact. The specific morphology of the hip varies with age, race, gender and developmental changes [8].
2.1.2 Articular Cartilage Composition and Behaviour

Articular cartilage is a biphasic soft-solid tissue less than 4mm thick comprised of an interstitial fluid phase and a solid phase that act in tandem to dampen and distribute loads between bones in joints, both of which are incompressible in isolation (Figure 2.2) [9] [10]. The solid phase is an amalgam of collagen fibrils, cartilage cells (chondrocytes), and negatively charged proteoglycans. Collagen fibrils are extremely stiff in tension and are the components of cartilage primarily responsible for resisting tensile strain, but do
not provide meaningful support of compressive loads. The solid phase is highly porous, with effective pore size in the range of 2.0 to 6.5 nm and porosity values as high as 0.75 to 0.8 [11]. Cartilage is an avascular tissue that receives nutrients through perfusion of synovial fluid through the solid phase structure.

Figure 2.2 Microscopic composition of articular cartilage.

Cartilage can be separated into four distinct zones, though in reality these zones represent
a single continuum with material properties that change with depth (Figure 2.3). In the superficial zone, porosity and fluid content is highest and collagen fibres are oriented parallel to the articular surface, allowing for high fluid flow. In the transitional zone directly below, collagen fibres have a random orientation and fluid flow and porosity is significantly decreased. Collagen fibres in the deep zone are oriented perpendicular to the subchondral bone. The deep zone has a low fluid content. Between the cartilage and the subchondral bone is a transition layer that has mechanical properties common to both cartilage and bone, and due to its high stiffness in relation to cartilage (approximately 10~100 times [12]), is not typically associated with cartilage deformation in simulations. The superficial zone, the transitional zone, and the deep zone comprise approximately 10%, 60% and 30% of the cartilage depth, respectively [13].

The porosity of cartilage allows for high fluid flow during compressive loading. Fluid pressure gradients are created in the solid phase by the negatively-charged glycosaminoglycan chains (GAGs) attached to the proteoglycans during the initial loading, so that in the initial loading the fluid phase supports the majority of the load. During this time, friction between cartilage layers is substantially reduced by biphasic lubrication. As compressive loading continues, interstitial fluid flows from out of the cartilage primarily at or around the contact area in the superficial zone [14]. At equilibrium, all of the interstitial fluid has been squeezed from the cartilage so that the entirety of the load is borne by the solid phase. At this point, the coefficient of friction remains low due to boundary lubrication caused by components of the synovial fluid adhering to the cartilage surface [15]. When the load is removed, the cartilage swells back to its initial undeformed state because of expansion of proteoglycan molecules that
create hydrostatic pressures, drawing fluid back into the tissue. Due to the effect of the solid and fluid components within cartilage acting in tandem during deformation, it is a tissue that exhibits highly nonlinear characteristics such as creep, hysteresis and viscoelasticity.

The most common mechanical tests of articular cartilage in the literature are confined compression tests, unconfined compression tests, and indentation tests. Unconfined compression consists of sandwiching the cartilage between parallel plates which are used
to apply compressive forces, while the free sides of the cartilage allow for fluid flow out of the tissue. In confined compression the cartilage is restricted by an impermeable chamber on the sides and bottom while it is compressed by a porous plate at the top. In indentation tests, a circular or spherical indenter is pressed into the cartilage surface to create localized strain and the resulting stresses and reaction forces are measured.

2.1.3 Femoral Acetabular Impingement

Abnormal bony alterations in the form of osseous growths on the femoral head or the acetabular rim can result in wear and degradation of articular cartilage within the acetabulum by causing supraphysiological biomechanical movement and loading. These protrusions cause impingements in a condition known as Femoral Acetabular Impingement (FAI), which is considered a potential initiator of osteoarthritis, characterized by mechanically induced loss of thickness of articular cartilage resulting in severe pain and discomfort. Osteoarthritis is the primary reason for the requirement of total hip arthroplasty. By the age of 55-65, up to 85% of the population will have some degree of Osteoarthritis [12], although it is possible that young adults may develop the condition when exposed to repeated and extreme motion that cause an abutment conflict between the femur and the acetabular rim [16].

FAI may be the result of repetitive stress caused by deformities separated into two categories: cam impingement and pincer impingement. In the case of cam impingement, the normally near-spherical femoral head has developed a deformity that disrupts the normal movement of the femur by contacting with the acetabular rim (Figure 2.4 centre), causing increased stress and damage to the articular cartilage in that region. Pincer
Impingements are the result of an osseous growth on the rim of the acetabulum, and similarly cause increased stress in the articular cartilage of the hip joint during movement (Figure 2.4 right). In most cases it is a combination of the two impingement types, resulting in damage symptomatic of each morphology.

**Figure 2.4** Impingement morphologies and contact areas due to osseous growths (red circles): normal condition (left), cam impingement (centre), and pincer impingement (right).

Identification of FAI cases is usually made through a combination of physical symptoms expressed by the subject, examination and 2D radiography [17], but abnormal morphology would also appear in computed-tomography (CT) images and magnetic resonance imagery (MRI).

### 2.2 Deformable Computer Modeling

#### 2.2.1 State of the Art

The field of deformable modeling can be considered to have begun with the work of Terzopoulos [18][19], who coined the term “deformable model” and was the first to introduce physically-based modeling into a deformable object to define the potential...
energy of a system of deformable splines, or active contours. Deformable splines are no longer used in practice due to their high computational cost in relation to subsequent methods, and consequently the necessity of modeling objects as non-volumetric, but these works paved the way for more complex, accurate, and computationally expedient procedures.

Introduced soon after the work of Terzopolous, mass-spring models simulate a deformable object with a series of point masses with interconnected springs (Figure 2.5). This method has been used for a variety of applications, including cloth [20][21] and surgery simulation [22][23][24]. Although it is easy to render these systems directly, unlike in the case of deformable splines, and they are computationally efficient, it is very difficult to achieve accurate deformation results using this technique. Because a mass-spring model is fundamentally composed of a non-volumetric system, incompressibility cannot be achieved. Furthermore, although nonlinearities can be incorporated into mass-spring models [21][25][26][27], these and more complex material properties such as anisotropy, viscoelasticity and inhomogeneity approximating real phenomena inherent in such materials as soft tissue can only be incorporated via involved, situation-specific parameterization techniques requiring the tuning of the model to experimental data [28][29][30].

The FEM is the most broadly used of any computer graphics method for deformable models for cases when accuracy is a necessity. In this procedure the continuum of an object is subdivided into numerous smaller elements and ordinary and partial differential equations are solved for the system to determine deformation (Figure 2.6). FEM has been used in a wide variety of applications in deformable modeling, with one of the first
Figure 2.5 Illustration of how a continuum can be divided into a series of mass-springs to simulate solid object behaviour.

examples being the work of Gourret et al. [31], who animated human skin deformation during a grasping task. Though this is an early work, it highlights the tradeoff exemplified by FEM: when accuracy is paramount, it is not yet reasonable for use in real-time scenarios due to the intense computations involved in solving the associated differential equations. The statement by Bro-Nielsen et al. [32] in their paper on surgery simulation illustrates this fact: “It doesn’t really matter whether the deformation that the surgeon sees in the virtual environment is accurate as long as it seems realistic.” As such, many attempts have been made to decrease the computational burden of FEM simulations of a deformable object.

Some of the most common approaches to increase the solution speed of FEM models involve some degree of model simplification. In the method of condensation [33][34][35]
Figure 2.6 Example division of material model into finite elements with colourization according to calculated stress. (For illustrative purposes only, not intended to represent an actual FEM discretization).

The internal nodes are removed so that only the surface nodes are considered. Adaptive approaches employ a hierarchy of meshes, with finer resolution replaced in areas where deformation is occurring [36][37]. Precomputation of a large number of the most likely deformed states allows for faster online simulations speeds, but only if the deformation conforms with those predefined scenarios [38][23]. Reduced order models store only the most relevant information of a complete FEM solution for use in creation of a simplified model where the most probable loading states are precomputed to achieve real-time interactivity [39][40]. Recently there has been increased interest in implementing FEM solutions on the Graphics Processing Unit (GPU) [41][42], but in most cases of soft tissue simulation, computational speed has been the focus rather than accurate reproduction of tissue response. Of course, there exist many simulations, especially of
soft tissues, that are not concerned with simulation speed but are instead intended to attain the most accurate results possible with current techniques [43][44][45]. One important improvement to the FEM technique unrelated to computational speed is Müller’s [46] proposed warped-stiffness approach employing a corotational formulation to remove the artifacts from linear deformations with simplified strain where higher order terms are ignored. Later plasticity and fracture were added to the simulation [47] (Figure 2.7).

![Figure 2.7 Incorrect FEM under large rotational deformations with inflated head (left) and correct deformation on the right (reproduced from [47]).](image)

In the Boundary Element Method (BEM) [48] only the surface of a mesh is discretized into elements. The calculations are carried from the interior of the model to its surface, allowing for a non-volumetric model to emulate a volumetric one. Despite this, BEM may in fact increase computation time due to the use of dense stiffness, mass and damping matrices. BEM has since been extended to inhomogeneity and anisotropy [49], but these solutions are far more complex and are based on linear elasticity.

The tensor mass model (TMM) was introduced by Cotin et al. [50] as a linear elastic solution for surgery simulation. Like FEM, this technique is based on continuum
mechanics, but unlike FEM which derives global interactions among all elements during each time step, TMM models iteratively solve the Newtonian laws of motion at each node. This allows TMM to exhibit much more realistic volumetric behaviour than mass-spring models while maintaining a timely solution. The initial formulation was extended by Picinbono et al. [51] for geometric non-linearity and large deformations, and later viscoelasticity was included [52][53]. Incorporating non-linearity and incompressibility greatly reduces computational speed, and many properties of soft tissue have yet to be considered in a TMM framework.

Many other methods of simulating deformable solids have been developed, including, but not restricted to: chainmail [54], the natural element method (NEM) [55], the point collocation-based method of finite spheres[56][57], and oriented particles [58].

### 2.2.2 Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics (SPH) is a meshless, particle-based method that was originally developed by Gingold and Monaghan [59] in 1977 to simulate stellar formation. It offers an attractive alternative to more conventional deformable models used to simulate soft tissue due to its inherent speed and its capacity to represent real physical phenomena in a precise manner. The inspired invention through which SPH is capable of representing a continuum with particles is known as a smoothing kernel, a function applied to the physical equations that has finite support centered at each particle, usually represented by a Gaussian curve (Figure 2.8). Rather than in the case of FEM where the computational solution of each element is dependent on every other element in the simulated object, the finite support domain around each particle enforced by the
smoothing kernel in SPH allows the computational burden to be reduced to only those particles considered to be in that particle’s neighbourhood, thereby considerably increasing simulation speed. The paper by Monaghan [60] provides a good primer to the study of SPH.

Figure 2.8 Smoothed particle hydrodynamics smoothing function with Gaussian support in a 3-dimensional domain. \( W(x_i - x_j, h) \) is the smoothing kernel function. \( i \) is the current particle. \( h \) is the smoothing length. \( x_i \) and \( x_j \) are the positions of the particle and its neighbour, respectively.
Since its introduction, SPH has found most frequent use in fluid dynamics simulations with an underlying physical dependence on the Navier-Stokes equations (Figure 2.9). The paper by Müller [62] provides a clear example of this type of formulation. It provides a method to simulate fluid flow at interactive rates, complete with a method of stabilizing the free surface of the fluid by surface tension forces. Clavet [63] adapted the SPH simulation to viscoelastic fluid. Lenaerts et al. [6] extended the method for the case of porous flow with Darcy’s law. Bao [64] and Raveendran [65] proposed means of enforcing incompressibility.

**Figure 2.9** Still frame of SPH fluid breaking dam simulation using the procedure implemented in this research, with visualization by S. Leblanc.

Various solutions have been proposed to represent solid boundaries in SPH. The simplest (albeit the non-momentum conserving) method is that of reflecting particle velocities upon crossing a boundary. There have also been attempts to use false “mirror” particles as particles approach boundaries, and boundaries composed of particles
supported by so-called “virtual” particles, with one recent example of the latter being that by Adami [66].

SPH is simulated in a Lagrangian framework where each particle carries its own properties with it through the continuum. This is beneficial for the application of SPH toward interactive simulations because it allows for each particle’s computations to be parallelized on the GPU, vastly increasing the speed at which solutions to the underlying physical equations are completed. Shifting the computational burden from the Central Processing Unit (CPU) to the GPU in SPH is now common. Two examples are the works by Goswami [67] and Zhao [68], each of which provide unique algorithms to streamline the SPH fluid formulation on the GPU.

Research into simulation of soft biological tissue with SPH has been relatively sparse. Müller [69] employed a fluid SPH formulation to represent blood flow through an artery with arterial walls composed of a tetrahedral mesh (Figure 2.10), while Qin [70] used a similar technique with walls comprised of mass-springs. Mesit and Guha [71] attempted a similar method for representation of a soft body composed of a mass-spring shell with gaseous internal pressure enforced by SPH particles. Work on deformable solids with SPH began with the work of Desbrun and Cani [72], but several other approaches have since been devised that allow for the introduction of elasticity theory into the SPH framework, which have made the requirement of a mass-spring shell to contain SPH particles unnecessary. Gray et al. [73] adapted such theory to the testing of bending, fracturing phenomena, and high impact tests, as did Cleary and Das [74] and Bonet [75]. In what is possibly the singular case of representing soft tissue using SPH in an elastic
body formulation, Hieber at al. [76] employed elasticity theory in the representation of a virtual liver, but the simulations were limited to small deformations in two dimensions.

![Figure 2.10](image.png)

Figure 2.10 Blood flow out of an injured artery. Blood is represented by SPH fluid particles and arterial wall is composed of a tetrahedral mesh (reproduced from [69]).

The work by Solenthaler et al. [77] represents a significant step forward when considered for the accurate simulation of cartilage using SPH. It provides an approach to derive an approximation of the Jacobian of the deformation field, but unfortunately the solution is prone to inaccurate deformations as rotations are falsely considered by the model to be strains, which results in forces that constrain objects in their ability to rotate. The work by Becker et al. [5] solved this problem by using a polar decomposition-based corotational approach to extract the rotation matrix from the deformation matrix of each particle neighbourhood so that only true strains are considered for elastic force calculations. This enables simulations involving large deformations with a range of elastic parameters for a rotating object to be implemented. It is now feasible to create realistic representations of physical materials based on more complex constitutive models.
2.2.3 Cartilage Computational Models

Since the biphasic theory proposed by Mow et al. [10], most mathematical models created for the purpose of accurately defining cartilage deformation have included incompressible solid and fluid phases in their formulations. A number of considerations arose with the investigation of compression models, particularly the difficulty in preparing an experimental specimen of cartilage for comparison that was perfectly cylindrical to the confining walls. Furthermore, this early model was isotropic and homogenous so its applicability to all possible loading scenarios was limited. The biphasic framework was subsequently used by Lai and Mow [78] to explore the effect of strain on the permeability of cartilage and to experimentally determine that the transient stress and stiffness of cartilage was dependent on strain rate. Mak [79] made a significant contribution to the field with the introduction of the biphasic poroviscoelastic (BPVE) mathematical model for articular cartilage, that incorporated fluid flow-dependent and fluid flow-independent qualities to simulate viscous effects that were found to contribute greatly to loading response in compression testing. After exploring indentation testing [80], the next major addition to cartilage models was again proposed by Lai and Mow [81]. Known as triphasic theory for its addition of ionic chemical interactions to the biphasic formulation, this was the first method to account for swelling behaviour due to osmotic pressure from charged particles, i.e. the glycosaminoglycan chains of the proteoglycans.

One of the concerns with the biphasic models proposed by Mow et al. was that it was isotropic, and as such was unable to predict cartilage behaviour in unconfined compression. The first attempt to rectify this problem was through the transversely
isotropic model introduced by Cohen et al. [82][83], which effectively stiffened the lateral plane during compression tests to simulate all collagen fibrils being oriented in the same direction, while the material was considered isotropic in other directions. The constitutive equation of the transverse isotropic model is given by:

\[
\begin{bmatrix}
\sigma_{11} \\
\sigma_{22} \\
\sigma_{33} \\
\sigma_{12} \\
\sigma_{13} \\
\sigma_{23}
\end{bmatrix}
= \begin{bmatrix}
\frac{1}{E_T} & -\frac{v_{TT}}{E_T} & -\frac{v_{LT}}{E_T} & 0 & 0 \\
-\frac{v_{TT}}{E_T} & \frac{1}{E_T} & -\frac{v_{LT}}{E_L} & 0 & 0 \\
-\frac{v_{LT}}{E_L} & -\frac{v_{TL}}{E_L} & \frac{1}{E_L} & 0 & 0 \\
0 & 0 & 0 & \frac{1}{G_L} & 0 \\
0 & 0 & 0 & 0 & \frac{1}{G_L} \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}^{-1}
\begin{bmatrix}
\varepsilon_{11} \\
\varepsilon_{22} \\
\varepsilon_{33} \\
\varepsilon_{12} \\
\varepsilon_{13} \\
\varepsilon_{23}
\end{bmatrix}
\]

\[G_T = \frac{E_T}{2(1 + v_{TT})}\]  \hfill (2.2)

where \( \sigma \) are the component stresses, \( \varepsilon \) the component strains, \( E \) the Young’s modulus, \( G \) the shear modulus, and \( v \) the Poisson’s ratio, with the subscripts \( L \) and \( T \) standing for the longitudinal and transverse directions, respectively. The conewise linear elasticity model by Soltz [84] is another adjustment to the constitutive formulation of biphasic linear elastic theory; however, unlike the transversely isotropic model it also incorporates compression-tension nonlinearity.

A more rigorous approach to the simulation of collagen fibrils was proposed by Soulhat et al. [85], by explicitly defining a collagen network for simulation that was
considered in loading scenarios to resist tension only, unlike previous models such as the transverse isotropic. Li et al. [86] extended the work of Soulhat into the biphasic fibril reinforced poroelastic model. This model and those that followed with similar formulations are arguably the most often used in the present day for simulating the mechanical behaviour of cartilage in compression. The extracellular matrix was considered to be linear elastic, but the collagen fibres were considered to grow increasingly stiff with increasing strain, which is regarded as one of the primary sources of nonlinear behaviour in cartilage deformation. Fibril-reinforced poroelastic models are capable of accurately simulating loadings in tension, unconfined and confined compression, and indentation tests.

Although not including the fibrillar network, DiSilvestro [87][88] successfully showed that the intrinsic viscous effects of the non-fibrillar network were necessary to obtain more precise results with the biphasic viscoelastic (BPVE) model for unconfined and confined compression, as well as indentation tests. Huang et al. [89] merged the conewise linear elastic model of Soltz and the biphasic poroviscoelastic model of Mak to similarly affirm the necessity of accounting for intrinsic viscous effects in the solid phase.

The basic fibril-reinforced poroelastic model has been extended in many and varied ways. Korhonen et al. [90][91] used the model to examine the equilibrium response of cartilage as well as proteoglycan depletion. Laasanen [92] employed the model in an investigation of the biomechanics of articular cartilage to verify the nonlinear compression-tension behaviour. Li et al. [93][94] investigated use of the model at higher strains, concluding that a hyperelastic function should be used for fibrils instead of the simple spring models proposed in their original formulation, as well including intrinsic
viscous effects at higher strain rates. The series of papers by Wilson et al. in 2005 [95], offered multiple advancements to the basic fibril-reinforced poroelastic model. They included swelling behaviour due to an ionic phase, much like that devised by Mow in the triphasic model, depth-dependent orientation of fibrils as in the arcade model of Benninghof [96](Figure 2.11), and fibril viscoelasticity implemented via a hyperelastic neo-hookean function dependent on the fluid fraction [97]. Lei et al. [98] determined fibril configuration based on distribution functions.

![Figure 2.11](image)

**Figure 2.11** Arcade model of Benninghof. Collagen fibres originate and terminate on subchondral bone and are horizontal in the superficial zone.

The previously defined models have not deviated significantly from their underlying bases in the ensuing years, but their usage, applicability, accuracy and validity have been explored through new methods and formulations of components, the most significant of
which will now be briefly summarized. The analysis by Huang et al. [99] performed tension-compression tests to determine the applicability of a hyperelastic function to finite deformation curves. Garcia et al. [100][101] similarly employed a hyperelastic function and proposed a new method of representing the viscohyperelastic contribution of the collagen fibres. Ateshian et al. [102][103] performed experiments to determine the effect of the anisotropy of the fibrillar matrix. Pierce et al. [104] proposed a fibril reinforced model that ensured isochoric deformations. Li et al. [105] extended their previous fibril-reinforced poroelastic model to three dimensions with fibre orientation, fluid flow and pressure varying in space. Korhonen et al. [106] varied the ionic concentration to determine changes in the tissue response in compression. Shirazi et al. [107] represented the anisotropic distribution of collagen fibrils by directional dependent volume fraction. Gu and Li [108] developed a model of the knee joint from magnetic resonance images with realistic fibre orientations to study small deformations. Finally, Seifzadeh et al. [109][110][13] proposed a non-linear biphasic fibre-reinforced porohyperviscoelastic model that was used to predict tissue response during high strains, to examine the material properties of engineered cartilage, and perform an optimization scheme to satisfy experimental indentation results.

It should be clear by the previous lengthy series of examples that the overarching trend of articular cartilage computational models appears to be to define these models in ever more precise detail in order to accurately predict tissue response in every conceivable scenario without regards to computational efficiency. Although precision is obtained to the sub-millimeter level, computational load is almost never reported. However, one expert in the field of cartilage simulation who recently published papers of
three-dimensional articular cartilage deformation analysis in the case of FAI was able to provide a general estimation on the order of four days or more for each of those simulations [111][112].
Chapter 3  Methodology

The method of SPH is founded upon the integral interpolant as given by Monaghan [59] in its generalized form as:

$$A(r) = \int A(r') W(r - r', h) dr'$$

(3.1)

where $A(r)$ is the function of the spatial coordinates $r$, $W$ is the smoothing kernel, and $dr'$ is a differential volume element. The solution of this interpolant through the use of integration methods achieves the desired result of treatment of the material properties as a continuum in a Lagrangian-based evaluation, as mentioned in the previous chapter. The generalized form of the integral interpolant allows for a range of possible simulation properties and methods with the use of varying basis functions and smoothing kernels.

In the current chapter, the proposed formulation of cartilage simulation using SPH is provided. Integration methods tested in the design of the simulation are outlined. Two neighbour search algorithms - a parallel octree approach and a grid-based cell-indexing method - are fully defined as they were implemented during the evolution of the program framework. A discussion of the method of SPH as applied to elastic solids is provided that will be used as the foundation of the cartilage simulation, as well as the method for accurate rotation handling as adapted to a fully parallelizable algorithm. A variety of
unique extensions to the elastic solid SPH framework for the accurate simulation of cartilage properties are provided. The proposed method of boundary enforcement to prevent penetration of the elastic solid with rigid objects is illustrated. Finally, the overall implementation of the proposed framework is discussed in detail as it relates to its particular coding requirements in C++ and nVidia’s CUDA parallel programming platform.

3.1 Integration Methods

To determine the properties of the SPH particles as they evolve through time, each loop of a simulation must represent a fraction of that time during which the representations of the associated physical quantities must be updated according to predefined parameters. Practically, the solution of the integral equations during each of these so-called time steps produce approximations to the integral interpolant. Appropriate selection of algorithms for the evaluation of particle parameters as they are updated is therefore a non-trivial consideration as it has a direct impact on not only the accuracy of the solution during each time step, but also for the stability of the simulation as a whole. These integration methods for the solution of the SPH integral interpolant may be divided into two broad categories: explicit, where the solution of each time step is approximated to a future point in time based on the current conditions, and implicit, where the current values are generated in an iterative manner to converge on a solution that deviates to minimization between iterations. For the proposed framework, only explicit integration methods will be considered as the necessity of iterating the solution in implicit methods requires
considerable computational work and as such was initially assumed to make the goal of a real-time simulation unrealizable with the available hardware.

Three different integration schemes were implemented for the solution of the partial differential equations. The simplest is known as explicit Euler, where the coordinates $r$ and velocities $v$ are updated according to:

$$r_{t+\Delta t} = r_t + \Delta v_t \quad (3.2)$$
$$v_{t+\Delta t} = v_t + \Delta a_t \quad (3.3)$$

where $a$ represents the acceleration determined through calculations in the current time step. The positions and velocity components at the subsequent time step, represented by the subscript $t + \Delta t$, are updated in parallel, and hence lose their dependence, which reduces the inherent stability of the simulation.

One method that introduces dependence between particle properties without introducing significant additional computational burden is known as the Leapfrog scheme, which is defined as:

$$v_{t+1/2\Delta t} = v_{t-1/2\Delta t} + \Delta a_t \quad (3.4)$$
$$r_{t+\Delta t} = r_t + \Delta v_{t+1/2\Delta t} \quad (3.5)$$

As can be seen in the previous equations, the updated positions are now calculated based on the velocities calculated at half time step intervals. In effect, the velocity and the positions “leap” over each other in each time step, hence the origin of the name of the integration method. It should be noted that to initialize the integration scheme, the velocity a half step backward in time from the origin is required, which is simple to
approximate using the initial values of the velocities and accelerations:

\[ v_{t-1/2\Delta} = v_0 - \Delta t a_{t0} \]  

(3.7)

In many SPH formulations, particularly for the calculation of viscosity effects in fluid simulations, the values of the velocities precisely at each time step is required rather than at half intervals, in which case the midpoint values can be found as:

\[ v_t = \frac{v_{t-1/2\Delta} + v_{t+1/2\Delta}}{2} \]  

(3.8)

Because of its dependence between positions and velocities, the Leapfrog method has advantages in stability over the explicit Euler approach.

The third proposed approach is known as the Runge-Kutta method, where the complexity of the solution is dependent on the “order” of the desired scheme. In fact, the first order Runge-Kutta scheme is simply explicit Euler as outlined above.

The evaluation of the Runge-Kutta scheme is determined by solution of the system of differential equations over multiple stages within the interval of each time step which are then combined for a final solution. Coefficients weight the contribution from previous stages in the calculation of the current one according to what is known as a Butcher tableau.

\[
\begin{array}{c|cccc}
  & 0 & 1/2 & 1/2 & 1/6 \\
 0 & 0 & 1/2 & 0 & 1/3 \\
 1/2 & 0 & 1/2 & 0 & 1/3 \\
 1 & 0 & 0 & 1 & 1/6 \\
\end{array}
\]

**Figure 3.1** Butcher tableau for 4th-order Runge Kutta integration method.
For the present implementation, 4th-order Runge-Kutta was used (i.e. 4 stages), with the associated Butcher tableau of Figure 3.1 [113].

In SPH, the objective is to update the particle coordinates according to their velocities in each time step, so the 4th-order Runge-Kutta scheme in this case takes the form based on the above Butcher tableau of:

\[ r_i^{(1)} = r_i^n \]
\[ r_i^{(2)} = r_i^n + \frac{\Delta t}{2} v(t^n, r_i^n) \]
\[ r_i^{(3)} = r_i^n + \frac{\Delta t}{2} v(t^n + \frac{\Delta t}{2}, r_i^{(2)}) \]
\[ r_i^{(4)} = r_i^n + \Delta t v(t^n + \frac{\Delta t}{2}, r_i^{(3)}) \]
\[ (3.9) \]

and the final updated coordinate positions are given by:

\[ r_i^{n+1} = r_i^n + \frac{\Delta t}{6} \left[ v(t^n, r_i^n) + 2v(t^n + \frac{\Delta t}{2}, r_i^{(2)}) + 2v(t^n + \frac{\Delta t}{2}, r_i^{(3)}) + v(t^n + \frac{\Delta t}{2}, r_i^{(4)}) \right] \]
\[ (3.10) \]

While not iterative towards ultimate convergence as in the case of an implicit scheme, 4th order Runge-Kutta should theoretically converge to a more accurate solution of the governing partial differential equations than either the explicit Euler or Leapfrog approaches.

Although Runge-Kutta increased the accuracy of the solution, the extra computational work required in reevaluation of the integration multiple times resulted in each loop requiring four passes. In other words, it did not decrease the amount of actual time necessary to evaluate the same simulation time. For this reason, Leapfrog is proposed for
use in the final implementation of the program as it provides acceptable stability and maintains the desirable simplicity of the explicit Euler approach.

3.2 Neighbour Search Algorithms

As previously mentioned, the inspiration of the SPH method is its ability to represent a continuum through the interaction of a particle with only those particles in its neighbourhood. As such, when working towards a real-time simulation it is imperative that the method for determination of those neighbours is suitably swift. Therefore, a method must be established that reduces the number of particles included in the neighbour search from the naive value of $N^2$. An octree algorithm provides a means of reducing the size of the searched area by breaking down the volume space into ever decreasing levels depending on the number of particles found in neighbouring cells in a hierarchical manner. Alternatively, a grid-based approach such as the cell-indexing method outlined below uses a predefined number of cells in a grid and performs a simple search of all neighbouring cells for neighbour particles. Both of these algorithms as they were implemented in the SPH formulation are provided as follows.

3.2.1 Parallel Octree Neighbour Search

The major difficulty in implementing an octree neighbour search in an SPH program intended to be performed in a parallel manner is its dependence on a serial format in its classical form. Traditionally, the tree is created by sequentially adding each particle to the tree, and when the number of particles in a cell exceeds a certain value, that cell is subdivided and a new cell hierarchy is created at a lower level with the previous cell as a
parent (Figure 3.2). Unfortunately, this does not work when extended to a parallel framework because the cell that each particle is assigned to is determined by the results from previous particles. To overcome this problem, a parallel octree method similar to that defined by Bédorf et al. [114] was implemented, the particulars of which for the current implementation for particle sorting, the tree creation and the tree traverse are given below.

![Diagram of a 3D octree](image)

**Figure 3.2** Example of hierarchical breakdown of nodes from the root in a 1-dimensional representation of a 3-dimensional octree. As levels increase, each of the preceding level’s cells are subdivided into eight more cells, unless a cell does not contain any particles (black) or contains less than a critical number (green). Lower levels cells are the children of the parent cells they are formed from.
3.2.1.1 Particle Sorting

The first step in the tree creation is to sort the particles based on their position so that they can be easily grouped together in cells, which is a deceptively complex procedure in a parallel format. Initially, particles are assigned three-dimensional Cartesian coordinates, x, y and z. In order to perform a sort, these positions must be converted from their three-dimensional 32-bit float values to single 64-bit integer representations through the application of Morton Coding, also referred to as z-indexing.

Before determining a particle’s z-index, it is first necessary to eliminate any decimals in their coordinate positions, which is done by multiplication of that position value by an appropriate integer value which can be later reversed when extracting positions from a z-index. For example, if the system has three-decimal accuracy then multiplying by 1024 should be sufficient to maintain all the relevant data contained in a particle position. These positions can then be shifted and interleaved to create their z-index via hexadecimal bitmasking operations. Figure 3.3 provides the particular coding for performing this operation in C++ for a 64 bit z-index based on three-dimensional Cartesian coordinates, purely for the purpose of clarity in the explanation. Essentially, the effect is that a pair of zeroes are inserted between each of the binary digits representing the x, y, or z coordinates and then each of those coordinates are offset to each other by a single binary digit before being interleaved into a single _int64 variable.

As in the method by Bédorf et al., a radix sort is performed on the z-indexes of the particles. While there are predefined libraries available that will perform a radix sort, a way was not found to apply them and maintain the association between the 64-bit z-index class member and to simultaneously sort the particles themselves. As such, it was
**Figure 3.3** Example C++ bitmasking operations provided for illustration of the method to extract the 64-bit integer z-index from three-dimensional Cartesian coordinates.

necessary to explicitly create the radix sort through a loop of all binary digits in the 64-bit z-indexes from the lowest to the highest bit. The method incorporates an all-prefix-sums computation similar to the one outlined in GPU Gems 3 [115]. Figure 3.4 provides an illustration of the radix sort algorithm from [115] followed by a brief description of its utilization, with encouragement to refer to the original source for more detail.

With the assumption that the integer values passed to the sort method only contain three binary digits for simplicity, the sort begins by performing a “split” on the least significant digit of each input, so that a new variable is set to 1 only if the applicable binary digit is 1. An array of false bits are created that are opposite to the array of splits. The false bits are then scanned for 1s, meaning that each time a 1 appears subsequently in the array a sum of the total number of 1s appearing until that point is placed in a new array, which gives the total number of false bits (0’s in the original input bit) as its last array member.
Figure 3.4 Radix sort example. Algorithm follows a top-to-bottom approach from input to output.

The value $t$ is determined from the following calculation: the input number, subtracted by the number of false bits until that point in the array, added to the total.
number of false bits. The array of $t$’s is passed to a function which returns the $t$ value if the input bit was 1, or the associated value from the false array if the input bit was 0, which is stored in array $d$. The input array is then reordered according to the values in array $d$. This procedure is repeated for each of the binary digits in the z-index of the particles from lowest to highest, resulting in a sorted array of particles from their lowest to highest z-index values. The scan in the radix sort algorithm can be performed as in [115], but the inclusive_scan function as part of the “thrust” library was found to be much faster in practice.

3.2.1.2 Tree Creation

At the start of the tree creation, a unique hexadecimal value must be assigned to each level of the octree. Nine levels were found to be sufficient for tree creation, but this may change depending on the number of particles, their density within the volume, and the volume size. Starting at the highest level (corresponding to eight cells, since a cell representing the whole volume is unnecessary) a loop through the particles is performed where their z-indexes are masked by the hexadecimal value according to that level and assigned to a new variable array associated with each particle. This loop was found to run faster in host code than CUDA due to its simplicity. A sort of the masks using the thrust library is performed on this array, which is possible because the array is not attached to a larger class, unlike the radix sort above, and a new array is created containing only the unique mask values for this level. A CUDA kernel is called to create the cells in the octree and assign particles, with each unique mask assigned a thread in a larger block of 256, which is the default number of threads in a block for all CUDA kernels used in the simulation. A number of cells is created equal to the number of unique masks at the
current level, each of which is assigned one of the masked values. A loop is performed through all particles for each cell, a linked-list is created from each cell to the first particle which shares the same masked value, and then from that particle to each subsequent one. During the loop, the total number of particles which share the current cell’s masked value is counted, and if the number of particles is less than some user-defined value, then this cell is defined as a leaf, or a termination point of that particle part of the tree. The linked list assigned to that cell is then traversed, assigning this cell as the parent to each of those particles. The above procedure is repeated for each level of the octree until each particle has been assigned to a leaf or the lowest level of the octree has been reached, whichever comes first.

All that remains in the process of tree creation is the act of linking each cell within the tree, or in other words, assigning child and parent cells to each cell where appropriate (Figure 3.2). This is accomplished through a pair of CUDA kernels. The first kernel determines what the mask would be for the current cell if it were a level below, and checks this against all cells in the level below it. The current cell is assigned as the parent to those cells which match this mask. It also extracts the Cartesian position information from the mask associated with that cell, which will be needed later during the neighbour search algorithm. The second kernel simply performs a loop through all cells and checks if the current cell is the parent while counting the total number of child cells. Because the cells were previously sorted, only the index of the first cell needs to be stored.

3.2.1.3 Tree Traverse

The traverse of the octree is used to perform the neighbour search for each particle, which is accomplished using a single CUDA kernel which can be separated into three distinct
loops. Each particle is assigned a thread in the kernel. The first loop is made through all levels of the tree from the bottom up, traversing from the parent cell of the current particle and setting each parent of that cell as the active cell for each level. The second loop assigns the current stack of cells as those at the highest level of the octree, and these will be checked in the initial run through of the third loop. The third loop performs the neighbour search itself, and beginning at the highest level, traverses the entire tree one level at a time.

In each level, the active cell is checked against each cell of the current stack to see if its distance is less than a value determined by the dimensions of the volume divided by a multiplier that increases by powers of two as the levels descend further into the hierarchy and cells decrease in size. If the check is passed and the cell is a leaf, the distance is checked between the current particle and all other particles within that cell to determine if that distance is less than whatever the interaction distance chosen for neighbours might be (which is the smoothing length in the current procedure). If the distance between particles check is passed, the particle is considered to be a neighbour, and its index is added to a link list of neighbours for the current particle. Alternatively, if the cell is not a leaf, that cell’s children are added to the stack to be used in the next level of the loop. After each level has been traversed, or no cells are assigned to the next stack, the neighbour search is considered complete for the current particle.

3.2.2 Cell Indexing

The second method of performing a neighbour search explored for use in the simulation is a simplified cell indexing approach based on the work by Onkerik and Durikovic
This method was implemented by a summer undergraduate student by the name of Sean LeBlanc under the supervision of the author. Using Axis-Aligned Bounding Boxes (AABB), the total volume space is subdivided into a three-dimensional grid of cells by the host, and the first particle in each grid cell is assigned to that cell with the remainder of the particles in that cell associated via a linked list. The particles are sent to a CUDA kernel where they are assigned individual threads and their 27 nearest AABBs are determined based on their current positions, which are the cells surrounding the current grid cell in three-dimensional space (Figure 3.5). A loop is performed through these cells and a distance check is performed between the current particle and the particles associated with each cell to determine if the checked particle should be added to the current particle’s neighbour list.

The cell indexing method presented above would at first investigation appear to be inferior to the previous octree method in total memory consumption, since unlike the octree approach, grid cells are created in areas where no particles are present, and in the case of a simulation where particles are confined to a small portion of the usable space this would infer a waste of resources in comparison. If the simulation was intended to be run in serial on the CPU, an octree method would likely be superior in its capacity to reduce consumption of memory resources. However, the octree method presented above requires such a complex system of calculations with multiple send and receive processes of data to the GPU and with many additional variable arrays to accompany those
those procedures that the cell indexing method requires far less resources and operates at a significantly greater computational speed. Without citing specific examples, the cell indexing method with a moderately-sized volume space and an average particle count of approximately 10,000 was found to result in a speed increase of approximately 5x over the parallelized octree method. Acknowledging that the cell indexing method above may not be as scalable to very large volume spaces as the octree approach, for modeling of the hip joint and cartilage to which the current procedure will likely be applied it is assumed that the gains in resources and speed in using cell indexing will be significant. Therefore, it is proposed that the cell indexing method presented above be used for the current procedure rather than the parallelized octree method.
3.3 Elastic Solids in SPH

The approach to elastic solid modeling in SPH presented here is based upon the work by Solenthaler et al. [77] and extended to perform in parallel on the GPU using CUDA. The choice of SPH kernel is fundamental to the stability and functionality of the proposed simulation, and an improper choice would likely result in the entire procedure to fail. As the goal of the current research is not to develop a new kernel for use in SPH simulations, it is more reasonable to use an existing kernel that has undergone rigorous testing in previously published research that can be assumed to function properly when applied to the SPH computations to be performed. The kernel to be used in these simulations was developed by Solenthaler for the specific case of elastic deformations:

\[ W(r, h) = c \frac{2h}{\pi} \cos \left( \frac{(r + h)\pi}{2h} \right) + c \frac{2h}{\pi} \quad 0 \leq r \leq h \]  

(3.11)

\[ c = \frac{\pi}{8h^4 \left( \frac{\pi}{3} + \frac{8}{\pi} + \frac{16}{\pi^2} \right)} \]  

(3.12)

where \( c \) is a constant that can be pre-calculated and \( W \) is equal to 0 for all other values of \( r \), which is the magnitude of the difference in positions of a particle and its neighbour.

Unlike the typical case of an SPH simulation developed to model the Navier-Stokes fluid dynamics equations, where the forces between particles are calculated based on cases of virtual pressure, the elastic solid procedure presented here is based upon the deviation between particles from their initial configurations. From this directly follows a procedure for calculation of forces from consideration of potential energy in the system. Strain energies are determined from the gradient of the displacement of the particles from
their initial positions, followed by calculation of the Cauchy stress using an appropriate constitutive model. In a perfectly elastic case where it can be assumed that there is no plasticity or permanent deformation in the system, these potential energies must return the particles to their initial positions to attain equilibrium. For simplification, it is assumed that the initial neighbourhood of each particles does not change, which is a reasonable assumption to make when considering perfectly elastic materials that are not exhibiting plastic change and are not excessively distorted. As the neighbour search accounts for a large portion of the total computational burden, performing that procedure only once per particle significantly increases the computational speed.

As in the case of fluid dynamics SPH simulations, the first calculation to be performed is the determination of the density of each particle from its initial neighbourhood, which is adapted from the generalized form of the SPH equation above to be:

\[
p_i = \sum_j m_j W\left(x_i^0 - x_j^0, h\right)
\]  

(3.13)

\[
\tilde{v}_i = \frac{m_i}{p_i}
\]  

(3.14)

where \( p \) is the particle density, \( \tilde{v} \) is the particle volume, \( x_i^0 \) is the position of the particle in its initial configuration, and \( m \) is the particle mass, which is the same for all particles. The subscripts \( i \) and \( j \) refer to the current particle and its neighbour, respectively. The densities can be precalculated based on the initial particle neighbourhoods and positions for the simple elastic case since the particle neighbourhoods do not change during the simulation.
In the SPH formulation, the local deformations \( \mathbf{u}_{ji} \) of each particle are calculated as:

\[
\mathbf{u}_{ji} = (x_j - x_i) - (x^0_j - x^0_i)
\]  

(3.15)

The gradient of the displacement field \( \nabla u_i \) is given the SPH approximation:

\[
\nabla u_i = \sum_j \tilde{v}_{ij} \tilde{u}_{ji} \nabla W(x^0_i - x^0_j, h)^T
\]

(3.16)

where the superscript T represents a matrix transpose. To relate the deformation tensor to the stresses in the material, the displacement field must be related to an appropriate choice of strain tensor. In this case, the non-linear Green-Saint-Venant tensor is proposed since it provides for a more accurate determination of the large deformations that would be expected to occur in simulations of soft tissue. The form used for the Green-Saint-Venant strain tensor \( \mathbf{E} \) is:

\[
\mathbf{E} = \frac{\nabla u_i + \nabla u_i^T + \nabla u_i^T \nabla u_i}{2}
\]

(3.17)

The Cauchy stress \( \mathbf{\sigma} \) can be related to the strain by a 4-th order (6x6) matrix, which in this case represents the constitutive equation for a perfectly elastic, or Hookean, material given by the variable \( \mathbf{C} \). The associated equations are:

\[
\mathbf{\sigma} = \mathbf{C} \mathbf{E}
\]

(3.18)

\[
\begin{bmatrix}
\lambda + 2\mu & \lambda & \lambda & 0 & 0 & 0 \\
\lambda & \lambda + 2\mu & \lambda & 0 & 0 & 0 \\
\lambda & \lambda & \lambda + 2\mu & 0 & 0 & 0 \\
0 & 0 & 0 & \mu & 0 & 0 \\
0 & 0 & 0 & 0 & \mu & 0 \\
0 & 0 & 0 & 0 & 0 & \mu
\end{bmatrix}
\]

(3.19)

\[
\lambda = \frac{\nu}{(1 + \nu)(1 - 2\nu)}
\]

(3.20)
\[ \mu = \frac{\varepsilon}{2(1+\nu)} \] (3.21)

The Lamé constants \( \nu \) and \( \varepsilon \) respectively are the Poisson's ratio and Young's modulus of the material, which represent the volume conservation and stiffness of the material respectively, and can easily be defined by the user at the beginning of a simulation.

Now that the Cauchy stresses for the particle has been calculated, the force exerted by particle j on particle i can be determined in the case of the Green-Saint-Venant tensor by:

\[ f_{ji} = \tilde{v}_j \left( 1 + \nabla u^T_i \right) \sigma_{ij} \] (3.22)

with the directional vector \( d_{ij} \) defined as:

\[ d_{ij} = \tilde{v}_j \nabla W(x^0_i - x^0_j, \hat{h}) \] (3.23)

Because Newton’s second law requires that any force exerted by a particle must be matched by a resultant force in the opposite direction to maintain momentum conservation, the final forces are calculated in a symmetric manner by:

\[ f_i = \sum_j \frac{-f_{ji} + f_{ij}}{2} \] (3.24)

### 3.4 Rotation Handling

The procedure for elastic solid simulation presented above functions well if the particle system is not subjected to rotations. However, because any change in the positions of particles are interpreted as strain in the above method, even if the only change is a simple rotation without any deformation of the particle system, this will falsely be interpreted as
strain and cause unrealistic and undesirable forces to return the particle system to its initial configuration (Figure 3.6).

![Image](image.png)

**Figure 3.6** Visualization of corotated rotation handling of particle neighbourhoods. Without the corotated approach, a rotation of the initial configuration is falsely interpreted as a displacement (centre). Inclusion of the corotated approach allows for rotations without creating additional forces (right).

To correctly handle rotations in the particle system, a corotational approach was proposed by Becker et al. [5] to back-rotate the particles to their initial orientations before force calculations are performed so that the “true strains” can be determined.

The deformation gradient of a deformed material can be separated into purely rotational and displacement parts. Therefore, the first step in isolating and eliminating the rotations from the force calculations is to determine the deformation gradient matrix \( A_i \) of particle \( i \). The calculation of the deformation gradient proposed by Becker uses the generalized form of the SPH equation to weight the contribution of neighbouring particles by the smoothing kernel:
The rotational part can be extracted from the deformation gradient using a number of different methods. In the cases of degenerated neighbourhoods or inversion, the always stable Singular Value Decomposition (SVD) [117] can be used to ensure stability. A deformation gradient matrix can be verified to be non-degenerated by checking that it contains all non-zero elements on the diagonal, and in the case of inversion, that the determinant of the deformation gradient matrix is greater than or equal to zero. Since degenerated neighbourhoods are assumed to occur only in planar models, for the majority of simulations polar decomposition can be used without concern for stability, although when each of these methods were implemented there was found to be little practical difference in computational complexity between the two methods. For brevity only the method of polar decomposition is shown, as that was the method used for all simulations in verifying the current procedure.

The first step in the polar decomposition is to multiply the transpose of the deformation gradient matrix by itself, i.e. $A_i^T A_i$, which yields a symmetric matrix. It is necessary to find its square root, which can be done by computing the diagonalization of this matrix. The diagonalization process is described in the following section. After that process is complete, the result is a diagonal matrix $D_i$ and an orthonormal matrix $V_i$ which contains the eigenvectors of $A_i^T A_i$ in its columns. The square root of $A_i^T A_i$ is simply a matrix multiplications of the orthonormal matrix $V_i$ and its transpose with the square root of $D_i$, which is computed as the square root of each element on the diagonal.
by:

$$\sqrt{A_i^T A_i} = V_i \sqrt{D_i} V_i^T$$ \hspace{1cm} (3.26)

It is now necessary to find the inverse of $S_i = \sqrt{A_i^T A_i}$, which is found through the trivial computation of:

$$S_i^{-1} = V_i \sqrt{D_i^{-1}} V_i^T$$ \hspace{1cm} (3.27)

The rotation matrix of particle $i$ can now be determined by:

$$R_i = A_i S_i^{-1}$$ \hspace{1cm} (3.28)

With the rotation matrix of particle $i$ in hand, the relations presented in the previous section must be updated to reflect the particle neighbourhood in a non-rotated orientation. This begins by back-rotating the local deformation $\bar{u}_{ji}$ so that the "true strain" can be determined by transforming the current particle displacements by the rotation matrix:

$$\bar{u}_{ji} = R_i^{-1} (x_j - x_i) - (x_j^0 - x_i^0)$$ \hspace{1cm} (3.29)

The computational burden and notorious pitfalls of calculating the inverse of a matrix can be avoided by noting that the matrix $R_i$ is orthonormal, and as such its inverse is simply its transpose.

Finally, the symmetric forces as calculated in Equation 3.24 must be returned to the orientations they were in prior to back-rotation by including the rotation matrix in this equation:

$$f_i = \sum_j \frac{-R_i f_{ji} + R_j f_{ij}}{2}$$ \hspace{1cm} (3.30)
3.5 Matrix Diagonalization

Unlike in the work by Becker, the current procedure has been optimized to run in parallel on the GPU using CUDA. For this reason, external libraries to perform matrix operations are not usable. While CUDA-based matrix operation libraries exist, these are created to run from host code and use parallel computing to expedite the solution of each matrix operation. This is not the case with the current procedure, where matrix operations must be performed from within threads of data that are being operated on while on the GPU. This requirement has a direct effect on the practical application of the method of polar decomposition (and similarly SVD). Performing the diagonalization of $A_i^T A_i$ in the above is nontrivial, the mechanics of which that were used in the current procedure will be briefly explained here.

The Jacobi method was chosen to perform the diagonalization, based upon that presented in Section 11.1 of the second edition of Numerical Recipes in C [118]. The idea behind the method is that a series of orthogonal transformations applied to $A_i$ will cause the off-diagonal elements to become vanishingly small. As suggested in the procedure given in Numerical Recipes in C, Jacobi rotations were performed until they reached the numeric precision limit of the data type used for the matrix elements in C++, in this case a float, since larger data structures are converted to floats when sent to the GPU via a CUDA kernel.
The rotation transformation matrix for a 3x3 matrix is of the form:

\[
P_{pq} = \begin{bmatrix}
c & 0 & s \\
0 & 1 & 0 \\
-s & 0 & c
\end{bmatrix}
\] (3.31)

for \( p = 1 \) and \( q = 3 \), where \( c \) and \( s \) are the cosine and sine of a rotation angle \( \theta \). The positions of the sines and cosines within the matrix are assigned according to the values of \( p \) and \( q \). For the example 3x3 matrix above, \( p = 1 \) and \( q = 3 \) means that elements in row and column 1 and 3 (and vice versa) will be occupied by the sine and cosine elements in that four-element pattern. Diagonal elements not containing a sine or cosine of the rotation angle are set to 1 and off-diagonal elements similarly are set to 0. The rotation angle is found from:

\[
\theta = \frac{a_{qq} - a_{pp}}{2a_{pq}}
\] (3.32)

where \( a_{qq} \) represents the value of the element in row \( q \) and column \( q \), and similarly for \( a_{pp} \) and \( a_{pq} \), noting that the matrix is symmetric so \( a_{pq} \) and \( a_{qp} \) will be equal after a rotation. The tangent \( t \) of the rotation angle is determined by:

\[
t = \frac{\text{sgn}(\theta)}{|\theta| + \sqrt{\theta^2 + 1}}
\] (3.33)

The cosine and sine of the rotation angle are:

\[
c = \frac{1}{\sqrt{t^2 + 1}}
\] (3.34)

\[
s = tc
\] (3.35)
A single Jacobi rotation of a matrix takes the form of:

$$B^{1} = P_{pq}^{T} \cdot B \cdot P_{pq}$$

(3.36)

where $B$ in this case is the matrix $A_{i}^{T} A_{i}$. Three Jacobi rotations are performed per “sweep”, with the first setting $p=0$ and $q=1$, the second with $p=0$ and $q=2$, and the third rotation having $p=1$ and $q=2$, noting that order of rotations is not important. In practice, it was found that the deformation gradient matrices were diagonalized within three to ten sweeps of rotations.

The transformations are accumulated through $n$ Jacobi rotations by:

$$V = P_{1} \cdot P_{2} \cdot P_{3} \cdot ... \cdot P_{n}$$

(3.37)

which is the matrix used in the previous section. The final diagonal matrix can then be verified by checking that the following relation holds true:

$$D = V^{T} \cdot B \cdot V$$

(3.38)

### 3.6 Articular Cartilage Property Implementation

Section 2.1.2 in the literature review chapter established the many diverse properties and components that together form the groundwork for the numerous FEM simulations that have been performed in the past in attempts to capture articular cartilage behaviour under different loading scenarios. While it may be possible to emulate some of the time-dependent behaviour of cartilage in a simulation through insertion of simple relaxation variables into the stress-strain relations, to obtain a more thorough and robust model one should include each of the primary components found in current material models,
reproduced in as exacting a degree as is allowable within the criteria and framework of the simulation.

Dissecting the afore-mentioned fibril-reinforced biphasic poro-viscoelastic models, three major parts are assumed to be necessary for an accurate reproduction of cartilage in SPH: a nonlinear non-fibrillar solid matrix, a collagen fibril network that extends throughout the cartilage volume, and a fluid component that has flow-dependence with the porous nature of the non-fibrillar matrix. That fluid component should also be capable of being exuded from the material under pressure or loading, and similarly to be retrieved from the region around the articular cartilage when that pressure is removed to restore the cartilage to its previous form, thus simulating cartilage relaxation phenomena. The resulting equation for the total stress is:

\[
\sigma_T = \sigma_{nf} + \sigma_f - \sigma_{fli}
\]  

(3.39)

where \(\sigma_T\) is the total stress from all components, while \(\sigma_{nf}\), \(\sigma_f\) and \(\sigma_{fli}\) are respectively the stress contributions from the non-fibrillar solid matrix, the fibrillar network, and interstitial fluid support.

Each of these components will now be defined as they are to be implemented into the framework of the SPH elastic solid simulation established in previous sections, and are intended to replace the prior Hookean stress-strain relation.

### 3.6.1 Neo-Hookean Solid Matrix

The solid matrix proposed for the current procedure is adapted from the compressible Neo-Hookean model proposed by Wilson et al. [95].
The governing equation for that model is given as:

\[
\sigma_{nf} = K \frac{\ln(J)}{J} I + \frac{G}{J} \left( F \cdot F^T - J^{2/3} I \right)
\]  

(3.40)

where \( J \) is the determinant of the deformation tensor \( F \), while \( K \) and \( G \) are bulk and shear modulus, respectively. The deformation tensor follows directly from the calculations to obtain the strain in the elastic solid equations, with the addition of the identity tensor, yielding the relation:

\[
F = \nabla u + I
\]  

(3.41)

The bulk and shear modulus are given by:

\[
K = \frac{\epsilon}{3(1-2\nu)}
\]  

(3.42)

\[
G = \frac{\epsilon}{2(1+\nu)}
\]  

(3.43)

whereas previously defined, \( \epsilon \) is the Young’s modulus and \( \nu \) the Poisson’s ratio.

This new stress-strain relation will function in the SPH simulation as a replacement to the Hookean relation even without the addition of the fibrillar network or biphasic fluid properties.

### 3.6.2 Collagen Fibril Network

A new method is proposed here for representing the collagen fibril network in an SPH simulation. There is a precedent for establishing fibre location according to element spacing in FEM in the work by Gupta et al. [119]. The reason for this is because an exact spatial representation of the fibre positions and orientations may not be available. Furthermore, it allows for the simplification of fibre placement within the SPH
simulation since fibres can be modeled as stress responses to deformations affecting existing SPH particles and their neighbourhoods rather than explicitly creating particles for their representation, resulting in a significant reduction of computational burden.

Fibres are oriented in the simulation according to the assumption that all fibres are at similar angles to the cartilage surface in each of the superficial zone, the middle zone, and the deep zone according to the values given in Table 3.1 [13]. The random orientation of fibres in the middle zone is considered to be 45° in most FEM simulations.

<table>
<thead>
<tr>
<th>Zone</th>
<th>Depth</th>
<th>Fibre Angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Superficial</td>
<td>0%-10%</td>
<td>0°</td>
</tr>
<tr>
<td>Middle</td>
<td>10%-70%</td>
<td>45°</td>
</tr>
<tr>
<td>Deep</td>
<td>70%-100%</td>
<td>90°</td>
</tr>
</tbody>
</table>

*Table 3.1* Collagen fibre orientation according to zone composition. Depth is measured from cartilage surface, and fibre angle is measured from a horizontally aligned cartilage surface.

For simplicity, it is assumed that all fibres are distributed between each particle and its neighbours. Changes in the distance between particles within a neighbourhood result in a change of strain and stress contribution from each individual fibre. Since it is assumed that fibres only strengthen the material model in tension, only positive strains are considered. The strain $E_f$ in each fibre is determined by:

$$ E_f = \frac{\left( x_j^0 - x_j \right)}{\left( x_j^0 - x_j^0 \right)} - 1 \quad (3.44) $$

Since in most cases the particles will not be oriented relative to each other in the same assumed fibre direction as defined in Table 3.1, an adjustment is necessary to the strain found in Equation 3.44. Using simple trigonometry, the angles between each particle and...
each of its neighbours in their current configurations, as well as their initial configurations at the beginning of the simulation, are calculated (Figure 3.7).

![Diagram showing the calculation of collagen fibre contribution based on orientation.](image)

**Figure 3.7** Method for calculation of collagen fibre contribution based on orientation. The horizontal green arrow represents the assumed collagen fibre orientation (superficial zone). The distance between particle i and neighbour j is represented by the light grey arrow. The stress contribution is adjusted by the ratio represented by the blue line.

This is because the particle neighbourhoods and their associated fibres may have changed orientation throughout the simulation if the cartilage has undergone deformation.

The new assumed fibre angle for the current zone is found by:

\[
\gamma = \gamma_0 + \left( \theta_i^0 - \theta_j \right)
\]

(3.45)
where $\gamma$ is the current assumed fibre angle, $\gamma_0$ is the original undeformed assumed fibre angle, while $\theta_j$ and $\theta^0_j$ are the current and undeformed angles between the particle $i$ and its neighbour $j$. This leads to a multiplication factor $M$ defined by:

$$M = \cos(\gamma + \theta_j)$$

which results in a one hundred percent contribution of the fibril strain to the stress calculations by fibres that are parallel with the assumed fibre direction, and conversely eliminates those fibres from the calculations that are perpendicular, with all other fibres lying in between.

The fibre strain $E_f$ must be assigned directionality to be used in the three-dimensional stress-strain relation. This is done by finding the unit vector defining the direction between particles $i$ and $j$ in the form of a 3x3 matrix:

$$u_f = \begin{bmatrix}
\frac{x_{jx} - x_{ix}}{||x_j - x_i||} & 0 & 0 \\
0 & \frac{x_{jy} - x_{iy}}{||x_j - x_i||} & 0 \\
0 & 0 & \frac{x_{jz} - x_{iz}}{||x_j - x_i||}
\end{bmatrix}$$

(3.47)

where $x_{jx}$ represents the x-coordinate of particle $j$, with similar representations for the other two Cartesian directions and for particle $i$. Shear strains are ignored so all off-diagonal elements are set to 0. This is a safe assumption since the fibres can be considered as one-dimensional structures for practical applications.

While using a viscoelastic model to define the fibres would be ideal (i.e. Wilson et al. [97]), because the current CUDA implementation is restricted to calculations performed
with floats, minute changes less than the numerical accuracy of this data type would cause huge, and unavoidable, stress variations that the simulation would not be able to compensate for and would quickly cause instability. Instead, the model suggested by Li et al. [86] should suffice to capture the fundamental behaviour of the fibril-network to the degree that is needed for the current procedure. The model is represented simply by a linear spring with stiffness \( E_0 \) in parallel with a nonlinear spring with stiffness \( E_1 = E_0 \varepsilon_f \) (Figure 3.8).

![Spring model representing a collagen fibre](image)

**Figure 3.8** Spring model representing a collagen fibre: a linear spring with stiffness \( E_0 \) in parallel with a nonlinear spring with stiffness \( E_1 = E_0 \varepsilon_f \).

After the inclusion of the previously defined multiplier and directional unit matrix, the resulting equation for the stress contribution by each fibre is:

\[
\sigma_f = -M_f u_f \varepsilon_f \left( E_0 + E_0 \varepsilon_f \right)
\]  

(3.48)

The contribution of the above equation is negative because it is defined based on the neighbour-to-current particle direction.

Although the proposed model captures the tensile-dependent behaviour of a fibril-network, because it is not based on a real spatial representation of collagen fibre
orientation the resulting stress contribution by the fibril network to the overall simulation is somewhat arbitrary. To account for this, the fibril network must be tuned to provide stress responses as close to published values as possible. This can be accomplished by changing the values of the individual variables attributable to the fibril-network, in particular the stiffnesses, or by simply running the code for the fibril-network loop multiple times until the desired level of stress response is achieved.

### 3.6.3 Porosity-Dependent Fluid Model

To incorporate fluid movement and support within the articular cartilage, a suitable method for representing the behaviour of that fluid must be selected. Acknowledging the necessity that this method be based upon the porosity of the non-fibrillar matrix, the work by Lenaerts et al. [6] based on Darcy’s law was chosen as the model to be adapted to the particular case of cartilage simulation. This method as it is to be implemented in the current procedure will now be defined, as well as a number of changes to its design so that it will function more closely with what is needed for the proposed simulation.

In the work by Lenaerts et al., porosity is treated in a macroscopic manner by assigning individual SPH particles as having a porosity value associated with it, and fluid mass is exchanged between particles whose pores are not completely saturated. Referring to Figure 3.9, the pore pressure (dark blue arrows) and the capillary pressure (green) create a balance of forces within each SPH particle causing fluid to flow into or out of the particle (light blue arrows). The amount and velocity of fluid flow is highly dependent on material properties, in particular the porosity (the void ratio), and the permeability (how
readily a fluid passes through). This model differs significantly from SPH fluid simulations that employ Navier-Stokes equations to determine their behaviour.

![Diagram](image)

**Figure 3.9** Example fluid mass exchange between SPH particles. Dark blue arrows represent pore pressure, green arrows represent capillary pressure and light blue arrows represent resulting fluid flow.

According to Rice and Cleary [120], the pore pressure $P_i^p$ is the pressure necessary to prevent mass exchange from a fluid-filled element that has been immersed in a reference fluid. This is difficult to relate to the current method as proposed by Lenaerts et al., since it employs the standard incompressible pressure equation from SPH fluid dynamic simulations, but it should nevertheless act to balance the capillary pressure in the system and force fluid mass out of particle neighbourhoods that have been deformed.

In the case of cartilage it should be assumed that the pores remain all or nearly saturated throughout the simulation, which necessitates some adjustments to the model.
The expression for the saturation $S_i$ of particle $i$ is given by:

$$S_i = \frac{m_{pi}}{\rho_{\text{fluid}} \phi_i V_i}$$

(3.49)

where $m_{pi}$ is the absorbed mass of the fluid in particle $i$, $\rho_{\text{fluid}}$ is the fluid density, $\phi_i$ is the particle porosity, and $V_i$ the particle’s current volume, which is calculated in Section 3.3.

Because the particles are considered to be fully saturated, and should remain that way in equilibrium since cartilage can be considered to be immersed in synovial fluid, the capillary potential $P^c_i$ which acts to draw fluid into a porous material via surface tension will only be non-zero in cases where the saturation drops below 100%:

$$P^c_i = k^c (1 - S_i)^\alpha \quad S_i < 1$$

$$P^c_i = 0 \quad S_i \geq 1$$

(3.50)

where $k^c$ and $\alpha$ are constants controlling the strength of the potential. This Equation 3.50 differs from the work presented by Lenaert et al. in that it now includes a boundary condition to prevent undefined behavior when the saturation of a particle exceeds the value of 1.

The form of the pore pressure is given by:

$$P^p_i = k^p S_i \left( \frac{\rho_i^S}{\rho_0^S} \right)^\gamma - 1$$

(3.51)

where $k^p$ is again a user-controlled constant, $\rho_i^S$ and $\rho_0^S$ are the current and original densities, and $\gamma$ is the user-controlled constant from the equation of state to enforce incompressibility.
In the original formulation by Lenaerts et al., the porosity is dependent on the local density of the medium in comparison to the initial density. However, with that interpretation pores can never be completely closed since no matter how much the material is deformed the porosity can never decrease to zero. This is not only difficult to conceive of from a physical standpoint, but it is also detrimental to the simulation of cartilage since it is necessary to be able to capture the properties of the cartilage once all fluid has been squeezed out of the material and the solid matrix supports the entirety of the load. It is therefore more logical to base the current porosity of the material on the current volume, according to the relation suggested here:

$$\hat{\phi}_i = \hat{\phi}_i^0 - \frac{V_i^0 - V_i}{V_i^0}, \quad \text{for all: } \frac{V_i^0 - V_i}{V_i^0} < \hat{\phi}_i^0$$

(3.52)

where $\hat{\phi}_i^0$ is the initial porosity of the current particle, $\hat{\phi}_i$ is the current porosity, and $V_i^0$ is the initial particle volume. Now if the ratio of the current volume to the initial volume is reduced to equal the initial porosity of the material (0.5 for example), the porosity will also become equal to zero, simulating all the fluid mass being exuded from that porous particle.

The porosity of cartilage is depth dependent, which can be incorporated into the current SPH procedure by the following relation:

$$\phi_i = \hat{\phi}_i \left(1 - \alpha_c \left(1 - \frac{z}{h}\right)\right)$$

(3.53)

where $\hat{\phi}_i$ is now the initial porosity at the articular cartilage surface, $\alpha_c$ is a material constant, $h$ is the height of the sample and $z$ is the current particle depth.2
The fluid density should be defined prior to beginning the simulation according to the properties of the synovial fluid, with the absorbed fluid mass dependent on the fluid density by the relation:

\[ m_{pi} = \rho_{fluid}^0 \phi_i^0 V_i^0 \]  

since the SPH particle should begin the simulation fully saturated. This gives rise to the eventuality (or requirement in the case of cartilage) of when a set of completely saturated particles undergo deformation, which is never touched upon by the previous work. Theoretically, each particle’s pores should expand under stress to accommodate the flow of fluid from its neighbours as their volume is reduced and thereby maintain the maximum saturation of 100%. While it should be possible to shift the positions of particles to expand the volume in each neighbourhood to achieve this effect, in practice altering the positions of particles in this way during an SPH simulation is a complex procedure that almost certainly results in loss of stability (as was verified through simulations). Instead, the solution proposed in the current procedure is to allow the saturation to exceed 100% and the associated pore pressure to increase along with it, with the saturation eventually returning to 100% at equilibrium due to that increased pore pressure.

An additional change needs to be made to the basic elastic solid simulation in order to incorporate fluid flow, and that is to update the density of the bulk material to reflect the addition of the fluid mass:

\[ \rho_0 = \rho_0^S + S_i \theta_i \rho_{fluid}^i \]  

(3.55)
The pore velocity $v_{pi}$ is the rate at which the fluid mass is diffusing from the material due to the pressure forces and capillary potentials weighted by the properties of the porous medium. Unlike the previous work, it was decided that the current procedure would base pore velocity explicitly on the difference between the capillary and pressure gradients between the current particle $i$ and its neighbour $j$. This was found to be more stable, more intuitive from a logical standpoint, and allows for the use of a single equation that performs a summation within the neighbour loop while still incorporating the gradients of the capillary potential and pore pressure:

$$v_{pi} = \sum_j -\frac{KV}{\phi_i \mu} \left( (P_j^p - P_j^c) - (P_i^p - P_i^c) \right) \nabla W(x_j - x_i, h)$$

(3.56)

where $K$ is the permeability of the material and $\mu$ is the viscosity of the fluid. The permeability has previously been defined in cartilage FEM simulations to be strain dependent. In practice this means the permeability will change with the porosity of the material by the relation [95]:

$$K = k_0 \left( \frac{1 + \phi_i^0}{1 + \phi_i} \right)^M$$

(3.57)

where $k_0$ is the initial permeability, $M$ is a positive constant, and $\phi_i^0$ and $\phi_i$ are the initial and current porosity.

The fluid mass diffuses between particles in a Eulerian manner rather than in a Lagrangian fashion typical of SPH simulations. This is because fluid particles are not explicitly created while the mass is contained within the porous material. This diffusion is defined by the differential:
\[
\frac{\partial m_{pi}}{\partial t} = \sum_j d_j V_j m_{pj} \nabla^2 W(x_j - x_i, h_j)
\]  

(3.58)

where diffusion coefficient \( d_j \) is a scalar that is the result of a dot product between the pore velocity and the unit vector of the neighbour-to-current particle direction to ensure appropriate contributions to the mass exchange by each neighbouring particle:

\[
d_j = v_{pj} \cdot \frac{x_j - x_i}{\|x_j - x_i\|} S_j^\beta
\]  

(3.59)

Here \( \beta \) is a user-defined constant to control diffusion, noting that in the current procedure this will have an additional effect to that originally intended by Lenaerts et al. in that it will increase diffusion with saturations greater than 1. An explicit Euler integration is performed during the update section of each time step to diffuse the fluid mass:

\[
m_{pi} \leftarrow m_{pi} + \Delta t \frac{\partial m_{pi}}{\partial t}
\]  

(3.60)

The entirety of the above procedure was implemented to define the fluid mass exchange between particles so that the current mass in each particle could be determined at each time step. The actual stress contribution from the fluid is defined by the current pore pressure:

\[
\sigma_{fl} = \eta P_i^p I
\]  

(3.61)

Again, this calculation differs from the work by Lenaerts et al. in that it uses the pore pressure derived from the previous SPH incompressible pressure equation instead of just the constant \( k^p \). This adjustment has been made to more accurately simulate increasing
pore stress with increasing deformation of the cartilage material, an effect which can be seen in many FEM cartilage simulations.

3.6.4 Fluid Mass Exchange With Cartilage Exterior

Explicit creation of fluid particles that are exuded from the cartilage material during deformation is not necessary, since unlike the work by Lenaerts et al., the concern in the current procedure is not with an ideal visualization but with numerical accuracy. It is beneficial to forego the creation of additional fluid particles because it saves greatly on computational resources. However, it also gives rise to the problem of how to simulate the fluid being exuded and subsequently being reabsorbed by the non-fibrillar matrix without doing so.

The proposed solution will be referred to as void particle instantiation, a new method particular to the above problem that will be detailed here. Void particles are essentially empty particles external to and surrounding the cartilage material that allow for fluid mass trapped within the cartilage to flow outward. In this way, it is much like the method presented by Lenaerts et al. for the creation of fluid SPH particles from a draining porous material. However, unlike that work, void particles are never filled and are never expected to behave like a fluid. They are essentially bottomless containers for the fluid within the cartilage to flow into until equilibrium is reached between the internal pore pressure and the capillary pressure of the SPH particles within the cartilage.

Void particles have several advantages over the creation of actual SPH particles to accommodate outward fluid mass flow. They only exist within neighbour lists and are without an associated index in the main program framework, so they are never considered
in loops through the complete particle list. They are not and cannot be visualized in the simulation space. Finally, their unlimited capacity for fluid mass means that additional particles never again need be created after the first group.

Figure 3.10 Void particle instantiation: The current particle position (blue) and the positions of its neighbouring particles (green) are checked to determine where void particles (white) should be placed. A) Four positions are currently unoccupied by particles, so four void particles are created at those positions. B) Seven positions are currently unoccupied by particles, so seven void particles are created at those positions.

The first step in void particle instantiation is to determine their desired positions (Figure 3.10). This is accomplished by performing a loop through the neighbours of each
particle and checking if neighbours already exist at predefined positions relative to the current particle depending on the initial particle spacing. Void particles are created at each position that is not already occupied by a neighbour and that neighbour is specified as a void particle by a boolean variable and the total number of neighbours is incremented. A simple check of this variable prevents void particles from being included in neighbour loops for other portions of the program such as the density calculations and the fibrillar network creation.

Void particles are only incorporated into two neighbour loops within the current procedure. The first is during the pore velocity calculation. For each void particle neighbour \( j \), equation 3.56 becomes:

\[
v_{pi} = \sum_j -\frac{KV}{\phi_j \mu} \left(-\left(P^p_j - P^c_j\right)\nabla W(x_j - x_i, h)\right)
\]  

(3.62)

This is the same as before, but with two exceptions: \( P^p_j \) and \( P^c_j \) are not included since the pore and capillary pressure of a void particle are assumed to be zero, and the volume of the void particle is assumed to be the same as the current particle. The changes are subtle but important. The result is that only the forces acting within the current particle have any effect on the amount of fluid mass exuded to or absorbed from a void particle. This means that a greater proportion of fluid mass will flow into a void particle from the current particle than into a neighbour that is experiencing increased pore pressure, for example, and the opposite is true for neighbours under negative pressure. This agrees with what intuitively should be expected in those scenarios. As well, particles that are being compressed or are resting upon a solid boundary do not employ void particles since boundaries are considered to be rigid and nonporous, and so will prevent fluid mass from
flowing outward. Note that Equation 3.62 requires the storage of the pore velocity of the void particle for future use.

Similar adjustments are made to the fluid mass diffusion differential equation:

$$\frac{\partial m_{pi}}{\partial t} = \sum_j d_{ij} V_i m_{pi} \nabla^2 W(x_j - x_i, h_j)$$  \hspace{1cm} (3.63)

where the diffusion coefficient $d_{ij}$ drops the saturation component since void particles are considered never to be saturated:

$$d_{ij} = v_{pi} \cdot \frac{x_j - x_i}{\|x_j - x_i\|}$$  \hspace{1cm} (3.64)

Finally, void particles require updating during the integration stage to maintain their positions and orientations relative to their associated deformable SPH particles. This can be accomplished simply by a loop that shifts their positions according to the same velocity and acceleration as that obtained for the current particle. Updating void particle positions without taking the current level of deformation into account may cause problems if the level of compression or extension is high, but for the simulation of articular cartilage within reasonable physiological limits this should not be a concern.

### 3.7 Rigid Body Boundary Enforcement

The boundaries of rigid bodies must be enforced to prevent penetration by deformable SPH particles. This is necessary for most SPH simulations, but is especially essential for the current procedure as a way must be found to simulate rigid indenters and surfaces to compress the cartilage for testing. A modified version of the virtual particle method [72] is proposed for use because it should ensure conservation of momentum within the
system, unlike simply reflecting deformable SPH particle velocities along their tangents, or creating mirror particles of approaching particles along the boundary.

![Diagram of boundary enforcement](image)

**Figure 3.11** Boundary enforcement: deformable SPH particles (blue) are prevented from penetrating the smoothing lengths of the boundary particles (green), which are given support by two layers of virtual particles (gray), resulting in a boundary that traces along the smoothing lengths (purple).

In the method of virtual particles proposed here, a single layer of boundary particles should be used along the rigid object boundary with two layers of virtual particles for neighbourhood support (Figure 3.11). The boundary particles are considered rigid and so do not update positions during the integration step, but can be moved during each time step according to the needs of the procedure. They apply a repulsive pressure force to any
deformable particle that penetrates its smoothing length, which is calculated via the pressure equations used in many SPH fluid dynamics simulations [60]. The density calculation for boundary particles is identical to equation 3.13 presented in the section on elastic solids in SPH. This is used to determine the pressure $P_i$ exerted by the boundary particle with the equation of state given in equation 3.54, reverted to its classical form of:

$$P_i = B \left( \left( \frac{\rho}{\rho_0} \right)^\gamma - 1 \right)$$

where $\rho_0$ is the original density of the boundary particle, $\gamma$ is a constant (set to 7 for the current procedure), and $B$ is a constant chosen so that the speed of sound is large enough to keep the relative density fluctuation $\frac{\delta \rho}{\rho} < 0.01$. For the current procedure $B$ can be set to $100 \frac{\rho_0}{\gamma}$. This ensures incompressibility, with the practical result that the penetration of any deformable particle within the smoothing length of a boundary particle is negligible.

The pressure force $f_{iP}$ experienced by the current deformable particle is a summation of all the boundary particles forces of which it is within the smoothing length. It is calculated via the equation:

$$f_{iP} = -\sum_j m_j \left( \frac{P_i + P_j}{2\rho_j} \right) \Delta W(x_j - x_i, h)$$

$P_i$ is the pressure of the current deformable particle, which is only calculated when it is within the influence of a boundary particle, although in practice this value does not change, since deformable particles do not update their neighbourhoods after the first loop in an elastic solid simulation. The boundary particles, on the other hand, must continually
update their neighbourhoods to determine if a deformable particle has entered their sphere of influence and so act upon them. However, since this only needs to be done for the single layer of boundary particles that form the boundary of any rigid object, it should not add a significant degree of computational burden to the simulation. The SPH kernel used in equation 3.66 is Desbrun’s Spiky Kernel [72]:

\[ W_{\text{spiky}}(r, h) = \frac{15}{\pi h^6} \left( h^2 - r^2 \right) \]  

(3.67)

\( W_{\text{spiky}} \) is equal to 0 for \( r > h \), where \( r = x_j - x_i \).

Note that boundary particles do not exert repulsive forces on each other or on virtual particles. They only act to repulse deformable particles that have entered within a smoothing length distance of them.

Because the pressure forces exerted by the boundary particles are directly dependent on the change of density of their neighbourhood, that neighbourhood must have enough support to ensure that the density is commensurate with the solid it is representing. It is for this reason that the two layers of virtual particles are included in the simulation to support the boundary, so that each boundary particle has a sufficient neighbourhood. The virtual particles themselves do not actively take part in any force calculations and are skipped over in the majority of the procedure. However, when the boundary particles move the virtual particles move with them to maintain their support.

### 3.8 C++ Program Framework

In this section it shown how all of the disparate pieces defined previously interrelate and are implemented within the framework of the C++ program devised to run them.
The C++ class “ParticleSystem” contains all the functions that will be called from the main file of the program, as well as the host (CPU) and device (GPU) versions of all of the program variables that will define the particle system and will be sent back and forth from the device during CUDA kernel calls. Because the variables are separate from the functions, only the variables that are relevant to each CUDA kernel call need be sent, decreasing the total memory required on the device. Allocation and deallocation functions are defined in the source code and contain the necessary instructions to transfer the variable data to and from the device, vastly reducing the number of code lines.

Figure 3.12 The algorithm used for simulation of deformable solids with the current procedure.
The program can be divided into two main sections as outlined by the algorithm provided in Figure 3.12. The first is the initialization stage that is only run at the beginning of the program. The second is the main program loop where each of the associated functions execute each loop.

The Maya Application Programming Interface (API) is an integral part of the process for the mesh generation and visualization. It is used to create the model meshes and populate them with Maya’s default particle system with positions regulated by a series of custom Maya Embedded Language (MEL) scripts that were created by Sean Leblanc for the specific purpose of visualizing SPH systems.

During particle initialization, the option is available in the program to define “mm resolution” so that particle systems created in the Maya workspace will have their positions scaled down by a factor of a thousand. This is because Maya does not provide the functionality necessary to easily create objects at the small scales necessary for cartilage. By defining millimeter resolution with the C++ program, the objects can be manipulated in Maya at the meter scale, while the C++ simulation treats the particles at the millimeter scale. At this stage, imported particle systems can also be defined as deformable, boundary or virtual. The position of each particle sorted according to its index and type can now be used in the remainder of the program.

Next, particles are sent en masse to the device where a cell indexing neighbour search is performed for the deformable particles. Error checks ensure that CUDA kernels behave properly each time they are called, which will also assist in determining the maximum number of particles that can be sent to the device, since an error will be returned if the
amount of sent data is too large. The density and volume of each particle is similarly obtained via a kernel call, followed by void particle instantiation if necessary for the simulation.

The main loop consists of five separate CUDA kernels that are called from a single function in the ParticleSystem class. A neighbour search is performed for boundary particles and their neighbourhoods are updated if deformable particles are detected within their smoothing length. Boundary particle densities are updated every loop, but if a biphasic material is being simulated the deformable particle densities are similarly updated, and it is in this same kernel that the values for pore/capillary pressure and saturation are obtained.

The second CUDA kernel is used to isolate the rotation matrix from the deformation. The third kernel calculates the collision forces exerted by boundary particles, as well as the pore velocity if required. The fourth kernel determines the strains, stresses, and associated forces between particles based on whatever material model is being used, including all the component stresses in the case of cartilage. The last kernel computes the symmetric forces acting on each particle, updates particle positions and velocities according to Leapfrog time stepping, and if necessary, integrates the fluid mass flow.

A write function designed by Sean LeBlanc is used to output the particle positions from each loop to .PRT files that can be imported into the Maya API for visualization. To reduce the number of files created and the number of frames to import into Maya, the option is provided to only output particle positions at specific intervals of the main loop (ex: every 100 loops). During this process, colorization of the particles to display in Maya can be performed to correspond to any variable associated with them. For example, if a
colourization of Cauchy stress is desired, the particles will display greater brightness for greater stress values. Similar to the write function, a block of host code for the output of particle variable values to an excel spreadsheet can be enabled to operate at specific intervals of the main loop.

Determination of the reaction forces to the application of pressure on the deformable object via a rigid body - an indenter, for example - is implemented by investigation of those particles that have previously been identified as being neighbours to a boundary particle. These boundary particles are exerting repulsive forces upon the deformable particles. Unfortunately, because these are pressure forces that are dependent on the amount of penetration into the boundary’s smoothing length and are not related to the actual force experienced by the deformable solid, another method must be used to make this determination. This is done by considering Newton’s second law. The particles repulsed by the boundary have a net force exerted upon them by their neighbouring deformable particles, which will be unbalanced by the degree that the boundary force is exerting on those particles. Therefore, all that is needed to determine the reaction force is a summation of the total unbalanced forces acting upon all the deformable particles that are neighbours to the boundary particles comprising the applied rigid body.

Each simulation is controlled by a series of macros in the header file “Vars.h”. For example, the cartilage properties or the simulation output can be enabled or disabled via simple commenting, and all constant variables used in the program can be easily adjusted here. Finally, a series of macros were created by Sean Leblanc to reduce the volume of coding required when performing matrix operations, which are considered to be faster
than equivalent functions since they are direct substitutions into the source code where indicated.
Chapter 4  Results of Elasticity and Cartilage Compression Experiments

The tests performed to validate the current SPH procedure can be divided into two categories. The first is the elasticity simulations that ensure that the fundamental framework of the SPH program is behaving as required. Two separate experiments were created to emulate those that were performed in the work by Becker et al. [5], each of which were designed to behave as realistically as possible in comparison to real materials if the corotation approach proposed in that work is functioning properly. The validation of these tests was primarily visual. After the elastic solid tests were completed and analyzed, the second step was to apply the current SPH procedure to the investigation of cartilage in indentation and unconfined compression experiments to determine if accurate numerical responses could be obtained relative to published experimental results.

4.1 Elasticity Simulations

Throughout the initial elasticity experiments, the smoothing length of the particles was set to 0.1 m and particles were arranged on a regular lattice at 1/2 smoothing length intervals. The Poisson’s ratios of the simulated materials were held constant at 0.46 to
simulate near-incompressibility, while the Young’s Modulus was varied between 1000 Pa and 1.5 MPa to ensure that the procedure was equally able to represent nearly fluid, jelly-like materials as well as higher ranges of stiffness as these would be necessary in the cartilage experiments to follow. A detailed evaluation of the maximum attainable time step size was not performed - such as that seen in many experimental procedures that use stability Criterion’s (for example the Courant-Friedrichs-Lewy [72]) based upon the highest foreseeable particle velocity, or the speed of sound - appropriate time step values were instead found with trial and error depending on whether stability could be maintained in the simulation. Such a criterion could be added into a future iteration of the design. As in the case of increasing the stiffness in SPH fluid dynamics simulations, it was found that with increasing Young’s modulus of a material, a decrease in the time step size was necessary to maintain stability, from as high as 0.5ms to as small as 0.001ms.

Two instances in time of the first test are shown in Figure 4.1. A large lower boundary and a second boundary perpendicular to the first along that boundary’s narrow edge were initialized with boundary and virtual particles as outlined in the procedure described in Chapter 3. A box-like configuration of deformable particles was generated in space approximately 1.5 smoothing lengths (0.15 m) above the lower boundary and artificially fixed to the vertical “wall” of boundary particles by disabling the ability to change positions of the nearest layer of deformable particles to that wall. Gravity (9.8 m/s^2) was then applied and the resulting motion of the deformable object was observed. Young’s modulus in the figure shown was set to 10 kPa, but tests were performed for as high as 1.5 MPa.

In cases of low Young’s modulus, the portion of the solid rod not affixed to the
vertical wall of boundary particles fell a short distance before coming to rest on the lower boundary. The lower boundary successfully prevented the deformable particles from any penetration into its surface, therefore validating the method used to enforce boundaries in the current procedure. The impact of the deformable particles with the lower boundary produced oscillations throughout the material that was highly dependent on the Young’s modulus of the material. At lower Young’s modulus values (1000 Pa), the rod exhibited significant oscillations, similar to a jelly-like material, which would continue for some time after the rod had come to rest before eventually subsiding in equilibrium. At higher Young’s modulus these oscillations were visibly dampened and subsided much more quickly. When the Young’s modulus was set to a high enough value, the solid rod would not impact the lower boundary at all, but would instead oscillate in space, thereby maintaining much of the original material’s shape despite the force of gravity acting on individual particles. At the highest tested Young’s modulus values the material would not visibly exhibit oscillations of any kind and would instead maintain its original shape and form, successfully simulating the behaviour of a material that was stiff enough to exhibit no deformation from the applied force of gravity.
When rotation handling was deactivated, the rod would unnaturally elongate as it was falling because particles would misinterpret rotation in a real material as an applied horizontal force. As can be seen in Figure 4.1 where the corotation approach to rotation handling was applied, there was only vertical extension in the particles of the solid rod, therefore behaving according to what would be expected of a real material.

![Figure 4.1](image.png)

**Figure 4.1** Falling wedge rotation test: t = 0 s and t = 0.5 s. Young’s modulus = 10 kPa.

The falling wedge test presented in Figure 4.2 was developed as an obvious proof that the method of corotation for rotation handling developed by Becker et al. and implemented in the current procedure was functioning properly. For this test, only the lower boundary was created and an upright wedge of deformable SPH particles was initialized and allowed to fall under the influence of gravity. With rotation handling active, the wedge falls over on to its side before coming to rest on the horizontal boundary, as can be seen in Figure 4.2. Without active rotation handling the wedge would simply “bounce” on its lower edge without tipping over because it was restrained in the horizontal direction by the elastic forces determined purely from the initial positions of
particles, hence the false strain forces as described in Chapter 3. Varying the Young’s modulus changed the behaviour as described in the solid rod test.

Tests in this section were performed on an Intel Core i7-3610QM CPU with 16 Gigabytes (GB) of Random Access Memory (RAM) and an NVIDIA GeForce 670M with 3GB of onboard memory. Elastic simulations were repeated many times over with a wide range of particle numbers to determine the relationship between simulated particle number and frame rate. The results of these tests from 1000 to 60,000 particles are shown in Figure 4.3. 60,000 particles was found to be the upper limit that could be successfully simulated with the current procedure. From a GPU with 3GB onboard memory to a second GPU with 2GB of onboard memory this maximum value decreased to 40,000 particles, which may indicate that the maximum number of particles in the higher range can be a linear relationship to the available memory of the GPU. As described in Chapter 3, each particle has a variable list associated by index number that is sent to a CUDA kernel for computation, and it was determined that reducing the amount of data (i.e. the memory required for these variables) increased the number of allowable particles in the simulation. A maximum of 60,000 particles was found to be the limit with the current framework, but this number could be increased if the amount of data required by these variables was somehow further reduced. Alternatively, a method could be found to transfer this data in partial “chunks” to be operated on by the GPU before being returned to the host memory for storage. Another option would be to use several GPUs or use a GPU with a larger amount of onboard memory. Fortunately, for the simulations being performed to validate the current procedure it was never necessary to simulate particle
numbers in the range of the maximum, but it may be required to increase the maximum for a whole hip simulation.

![Graph showing the relationship between the number of simulated particles and frame rate.](image)

**Figure 4.3** relationship between the number of simulated particles versus frame rate of elasticity simulations.

From examination of Figure 4.3 it is clear that frame rate decreases significantly with an increasing number of particles. The computational gain of employing CUDA over purely host code increases with increasing number of particles simulated. For 1000 particle simulations, CUDA performed at 257 frames per second (fps) versus 167 fps for the CPU, 1.54 times as fast. When the number of particles simulated is increased to 60,000, CUDA performs at 6.3 fps versus 2.6 fps for the CPU, 2.4 times as fast. This can be explained by the performance requirements of the current procedure as it transfers data. Each of these data transfers require a finite amount of time, effectively resulting in
computational overhead. At higher particle numbers with greater associated computational burden the time required to performed these data transfers accounts for a decreased percentage of the total simulation time.

The relationship between particle number and frame rate is not linear, but instead roughly follows a plot of $e^{-x}$. This is likely because the computational burden that is linearly dependent on particle number begins to account for a dominant percentage of the total. Finally, while the performance of these simulations might at first appear to be inferior to the published results of previous SPH research (see for example [63][122]), this would not be accurate as elastic solid simulations are comparatively far more computationally intensive than fluid dynamics simulations.

### 4.2 Cartilage Simulations

The elastic SPH procedure can be extended to the simulation of cartilage by the addition of the elements discussed in Section 3.6. The cartilage simulations must be validated for two different uses. The first is to verify that the areas of highest stress in the simulated cartilage can be visually identified. The second is to determine the reaction force to the indenter or compression plates during deformation and validate that against previously published material. Suitable experimental results must be found for comparison with the results of the simulations presented here. Three models were created for verification: two models to be used in indenter experiments and one for unconfined compression. The vast majority of experiments presented in the available research on cartilage compression behaviour examine the phenomenon of relaxation at long time scales after an initial strain. Perhaps the most often cited experiments are those by
DiSilvestro et al. [88], which present unconfined compression data and both porous and nonporous indenter results over a period of 1000 seconds. As can be seen in the data presented there, the reaction forces relax to less than 10% of their peak magnitude within that time frame. This effect is due to the flow of fluid mass out of the cartilage, meaning that in the instantaneous or short time condition (<5s for peak deformation) the pressure exerted by the fluid in response to compression supports the majority of the load. At the present time, this effect cannot be simulated with the current procedure since the stability of the simulation is dependent on the interactions of the SPH particles that comprise the solid material - corresponding to the non-fibrillar matrix - and not the fluid mass. However, the short time compression behaviour of cartilage can still be simulated effectively, as will be shown in this section.

Short time experimental results for validation were obtained from the work by Vidal-Lesso et al. [123] for the indentation experiments and Li et al. [93] for unconfined compression. It should be noted that for the compression experiments the comparison is made to the Li at al. FEM model results whereas for indentation only the experimental results by Vidal-Lesso et al. are considered. It should also be noted that the short time compression experiments performed by Vidal-Lesso et al. correspond to the loading of the human body at real physiological frequencies, as stated in that work, and as such are highly relevant to a simulation for preoperative investigation of FAI.

### 4.2.1 Indentation Experiment A

The first step in determining the validity of the procedure for cartilage simulation is to determine how close the results obtained through the simulation are to the experimental
case if the material is treated as perfectly elastic, since in the instantaneous case cartilage should be
have like an elastic material if relaxation phenomena are ignored. For this to be correct, the same experimental parameters must be used as determined by Vidal-Lesso et al. Indentation experiment A in the current investigation corresponds to the experiment performed on specimen 1 in that work. In each of the models used in the current procedure, the smoothing length is 1 mm and the particle interval is 0.5 mm. This means that the height of specimen 1 of 1.87 mm cannot be represented precisely and is instead rounded to 2 mm, or a depth of 5 particles. A higher density of particles could be used to more accurately represent the cartilage depth, but it is thought that if the current procedure were applied to a real hip model, an unreasonably large number of particles would be required.

A 3 mm diameter indenter composed of 246 boundary and virtual particles induces strain in the surface of a cylindrical cartilage plug of 8 mm diameter composed of 1624 deformable particles. The indenter is displaced by 0.5 mm at a rate of 0.21 mm/s, allowed to relax for 1.5 s, and then removed from the cartilage at the same strain rate for a total experimental time of approximately 6.26 seconds. Young’s modulus is set to 4.81 MPa and Poisson’s ratio to 0.45. Unlike in the work by Vidal-Lesso et al, the Young’s modulus is kept constant throughout the simulation. The material density is 1096 kg/m$^3$, yielding an individual particle mass of $1.5 \times 10^{-7}$ kg. Gravity is ignored and the bottom layer of particles is fixed in the simulation to prevent the material from sliding out from beneath the indenter since there are no frictional forces to prevent that from occurring. Due to the high stiffness values used in these simulations it was found necessary to reduce the time step size to $1 \times 10^{-5}$ seconds.
Next, the same simulation is performed for indenter experiment A with cartilage components active. Hereafter the biphasic poroviscoelastic model used in the current procedure will simply be referred to as biphasic for convenience. In the ideal case, experimental parameters determined from published research (ex. a Young’s modulus of 0.315 MPa for the non-fibrillar matrix as in Wilson et al. [95]) could be used to define the cartilage components and the results of the simulations would match the experimental by default. Unfortunately, because of the restrictions imposed by the current procedure on the stability, the fluid cannot support a high percentage of the load. However, in short time relaxation cases the observed results can be suitably matched by assuming that the fluid is pushed out of the cartilage material at a much higher rate.

To achieve this, the Young’s modulus of the non-fibrillar matrix was set to 3.0 MPa and the permeability was set to $3 \times 10^{-14}$ m$^4$/N s, which is much higher than the permeability given by Wilson et al. of $1.522 \times 10^{-15}$ m$^4$/N s. Increasing the permeability results in fluctuations in the fluid mass between particles and decreases the stability of the simulation, so a higher permeability than this is ill-advised. However, the simulation does not yet yield a high enough change in material stress to mirror the results of Vidal-Lesso et al.. To account for this discrepancy, the contribution of the pore pressure to the total pressure in the material was multiplied by the constant $\eta = 20$ in equation 3.55. This highlights why it was necessary to first verify the simulation with the perfectly elastic case; by changing the parameters in this way it would otherwise be simple to match any conceivable reaction results without any real basis for validation.

Depth dependent porosity was initially simulated with a void ratio of 0.85 at the cartilage surface, decreasing to 0.7 at the bone-interface. While the simulations did not
destabilize completely, the high porosity at the surface and associated greater load supported by the fluid mass caused undesired fluctuations in the fluid mass exchange. For this reason the base porosity was kept at a constant 0.7 decreasing to 0.55 at the bone interface. The fluid density was assumed to be the density of water for simplicity, \( \rho_{\text{fluid}} = 1000 \text{ kg/m}^3 \). The capillary and pore pressure constants were the same as that proposed in the work by Lenaerts et al., \( k^c = 150000 \), \( \alpha = 0.1 \), \( k^p = 50000 \), \( \gamma = 7 \), with the viscosity \( \mu = 0.01 \), and the diffusion constant \( \beta = 7 \). The collagen fibril stiffness constants were set as \( E_0 = 2.7 \text{ MPa} \) and \( E_c = 340.5 \text{ MPa} \), with a contribution of three loops. The full list of parameters used in these simulations is summarized in Table 4.1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \rho_0^\delta )</td>
<td>1096 kg/m³</td>
<td>( k^c [6] )</td>
<td>150000</td>
<td>( \beta [6] )</td>
<td>7</td>
</tr>
<tr>
<td>( \rho_{\text{fluid}} )</td>
<td>1000 kg/m³</td>
<td>( k^p [6] )</td>
<td>50000</td>
<td>( \alpha [6] )</td>
<td>0.1</td>
</tr>
<tr>
<td>( \varepsilon^* )</td>
<td>3.0 MPa</td>
<td>( \eta^* )</td>
<td>20</td>
<td>( m )</td>
<td>1.5 x 10⁻⁷ kg</td>
</tr>
<tr>
<td>( \varepsilon_{\text{elastic}}[123] )</td>
<td>4.81 MPa</td>
<td>( k_0^* )</td>
<td>( 3 \times 10^{14} \text{ m}^4/\text{Ns} )</td>
<td>( \Delta t )</td>
<td>1 x 10⁻⁵ s</td>
</tr>
<tr>
<td>( E_0[95] )</td>
<td>2.7 MPa</td>
<td>( \phi_i^* )</td>
<td>0.7</td>
<td>( h )</td>
<td>1 mm</td>
</tr>
<tr>
<td>( E_c[95] )</td>
<td>340.5 MPa</td>
<td>( \gamma [6] )</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \nu )</td>
<td>0.46</td>
<td>( \mu [6] )</td>
<td>0.01</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 4.1 Material parameters and values used in the cartilage compression simulations.*

*Tuned parameter due to stability restrictions*
The numerical results of both the elastic and biphasic simulations are presented in Figure 4.4 along with Vidal-Lesso’s experimental. The elastic simulation with the same simulation parameters as defined by Vidal-Lesso’s experiment produces a result that is strongly correlated to the experimental ($r=0.976$), with a peak reaction force approximately only 10% higher. The reaction force is initially lower than the experimental before becoming higher nearer to the peak, which is reasonable considering the load supported by the fluid in the experimental case is gradually decreasing as it flows out of the cartilage. As expected, the elastic simulation does not exhibit any relaxation phenomena. Conversely, the biphasic simulation as defined above is even
more strongly correlated to the experimental ($r=0.993$), with reaction forces that exhibit short term relaxation that are nearly identical to the experimental.

![Figure 4.5 Visualization of stress distribution of indentation experiment A.](image)

- **A**) Initial condition (0 s).
- **B**) Maximum strain (2.38 s).
- **C**) End of relaxation (3.88 s).
- **D**) End of simulation (6.26 s). (1624 particles)

The correlation is a measure of the linearity between data sets, and is determined here as a convenient method of approximating the relationship between the simulated and
experimental methods. It cannot account for differences in scale beyond the linear relationship, and it is likely a more suitable measurement could be presented via more advanced graphical methods using standard error, but as a cursory evaluation it is suggested here that correlation is sufficient for the current evaluation when combined with visual analysis of the resulting graphs.

The second requirement for the simulation is that one must be able to visually identify areas of increased stress in the cartilage. This is shown in Figure 4.5. At t=0 s, the indenter has not yet applied any pressure to the surface of the cartilage. At t=2.38 s the indenter has reached its maximum strain of 0.5 mm into the cartilage surface and the area of increased stress has spread from the point of deformation to particles nearby. At t=3.88 s the fluid mass has further diffused, spreading the stress more evenly throughout the material. This effect can be seen more clearly in Figure 4.9 for indenter experiment B. Finally, at t=6.26 s the indenter has returned to its original position.

An area of lower than equilibrium stress exists in the area where the indenter caused deformation. This is a reasonable result, because the fluid mass has yet to fully return to the cartilage at this location where it was displaced, corresponding to an area of negative pore pressure as the cartilage swells back to its original equilibrium condition. This corresponds to the results seen in Figure 4.6, which exhibits the changing fluid mass contained within a single particle in the vicinity of the indenter during the deformation. The fluid mass does not diffuse as smoothly as desired, which results from the high permeability causing fluctuations in the Eulerian fluid mass exchange.
The specific contributions of each of the cartilage components to the total material stress profile during the deformation is presented in Figure 4.7. It is an open problem how to compare stress components in such a simplified form. While the pore stress in Figure 4.7 is a scalar that is applied to the total stress of the simulation through multiplication with an identity matrix, the fibrillar stress is calculated as the average of the three principal directions of the stress contributions of the fibrils for each particle, which is then averaged again over all particles. In addition, the total material stress is based on the average of the three principal stresses corresponding to the three principal directions, ignoring shear stresses, and the equivalent non-fibrillar matrix stress is then computed simply by the subtraction of the pore and fibrillar scalars from this value. From the above it should be clear that Figure 4.7 and those like it should be used for illustrative purposes only, however it does provide a tool to analyze the trends experienced by components.
during deformation. Considering the resultant reaction forces are very well correlated to the experimental results, the inability of the scalar representations of the components to precisely match the force response results should not be a concern, because those are determined by their actual contributions and interactions and not by these simplified scalars.

Figure 4.7 Component stresses of biphasic simulation of indenter experiment A. (1624 particles)

In Figure 4.7, the contribution of the fluid mass to the total material stress in the form of pore pressure accounts for greater than 50% of the stress during the initial deformation. In other words, this appears to indicate the fluid pressure bears the majority of the load. In Figure 6 of the work by Li et al. [93], the fluid at strain rates exceeding 5%/s account for greater than 98% of the load. This is obviously not accurate for the
current procedure where the fluid is unable to support even as much of the load as indicated by Figure 4.7 without losing stability. However, the overall trend is that of increasing fluid stress during deformation that relaxes at the same rate as the force response, which is exactly what should be expected.

As relaxation occurs due to the flow of fluid mass out of the area of deformation, the non-fibrillar matrix can be seen to bear progressively more of the total stress. At the end of the experiment when the indenter has been fully removed from contact with the cartilage, there is a residual stress due to the non-fibrillar matrix balancing the negative pore pressure as it returns to equilibrium. The indenter displaces the central region of the cartilage, causing tensile stress to appear in nearby fibres in the non-axial directions, since that is how the fibres are oriented in the cartilage surface. It is stated by Li et al. “The fibril stress is identical in magnitude but opposite in sign to the radial stress of the non-fibrillar matrix.” While in the current procedure the Cartesian coordinate system is used, this statement still holds true. As can be seen in Figure 4.7, stress in the non-fibrillar matrix completely balances out the stress contribution from these fibres, but unlike in the work by Li et al. the pore stress no longer accounts for greater than 98% of the total material stress, which is why the non-fibrillar matrix stress is non-negative.

There are two questionable results that become apparent during examination of Figure 4.7. This first is the seemingly random accumulation of stress in the first 0.1 s of the simulation. This is again due to fluctuations in the fluid mass and can be ignored. The second problem is that the stresses appear to be an order of magnitude lower than what was presented in Figure 2 of Li et al. This discrepancy is likely explained by the method of calculating the stresses as explained above. Again, considering the resultant reaction
forces are very well correlated to the experimental results, it is thought that the disparity in the stresses present in Figure 4.7, and those like it in the forthcoming analysis, are very likely the result of the lack of a clear way of comparing these stress contributions with single scalar values rather than a problem with the underlying procedure. That aside, the overall trends in terms of stresses for each component appear to be correct.

![Figure 4.8 Fibril stress distribution of indenter experiment A at 2.38 s: A) front view, B) opposite view.](image)

Perhaps more instructive are the visual results of the fibril stress response at greatest deformation, presented in Figure 4.8A, which appear to be unevenly distributed to one side of the cartilage plug. There are two possible reasons for this to occur. This first is due to the lack of friction in the system. It was found that if the lowest layer of cartilage was not fixed during the simulation, the cartilage plug would slide out from under even the smallest applied deformation at a high rate. This indicates also that the indenter could not be directly centered over the cartilage mass. Secondly, the indenter and the cartilage mass itself are only as perfectly cylindrical as was possible to create given the restrictions
of the Maya particle population method. These two factors imply that when the indenter begins to deform the plug, it not only distributes that deformation slightly unevenly over the total cartilage mass, but there is no friction restraining the plug from deviating from its central position beneath the lowering indenter. Together this would cause the plug to “bulge” to one side, resulting in the uneven distribution of tensile strain in the horizontal direction. It should be noted that a low coefficient of friction is actually desired because in a real anatomical case there is very good lubrication between opposing cartilage surfaces within the acetabulum due to an array of mechanisms described in chapter 2.

The fibres are distributed in the horizontal direction in the surface only, while the two layers below are oriented at 45° to the horizontal, with the bottom two layers being vertical, which is why the stress is concentrated in the top layer and to a lesser extent the two beneath it. Figure 4.8B provides a view of the cartilage from the opposite side. It appears to support the hypothesis of a bulge to one side, as fibres are recruited in the middle layers to a greater extent due to increased deformation in the vertical direction.

A similar investigation of the pore pressure stress contribution is provided in Figure 4.9. Again, the distribution appears to exhibit the slightly uneven stress distribution of the previous figure, although not to the same extent, likely for the reasons given previously.

### 4.2.2 Indentation Experiment B

The results obtained with the biphasic simulation of indentation experiment A were tuned to behave as closely as possible with the experimental, and so are applicable only to that specific case, and do not validate the numerical accuracy of the procedure as a whole. To truly validate the biphasic simulations as proposed in the current procedure,
precisely the same material parameters as used in that experiment must be used in all other experimental cases to determine how closely they follow the experimental results. Essentially we have tuned the biphasic parameters in indentation experiment A for the remainder of the experiments. If those experiments also produce simulation results that closely follow the experimental, one can be cautiously optimistic that the biphasic procedure should be valid for the simulation of cartilage compression behaviour for the majority of cases.

![Figure 4.9](image.png) Pore stress distribution of indenter experiment A at 2.38 s: A) front view, B) opposite view.

Indentation experiment B corresponds to specimen 5 from the work by Vidal-Lesso et al., using the same experimental parameters as defined for indentation experiment A, with the exception of the Young’s modulus for only the elasticity simulation, which is adjusted slightly to 5.07 MPA to reflect the value as reported in that paper. As well, the cartilage depth is increased to 2.79mm, which for the current procedure is rounded up to 3 mm with an associated 7 particle depth. The cartilage is represented by 3206 particles
and the indenter is again composed of 246 boundary and virtual particles. The results can be seen in Figure 4.10. As expected, the reaction force in the biphasic simulation more closely matches the experimental results than the elastic material. Microsoft Excel 2007’s statistical analysis produced a correlation of $r=0.986$, whereas the biphasic simulation only has $r=0.981$. As stated previously, this is only a measure of the linearity between data sets. While the relationship between the elastic and the experimental simulations may be slightly more linearly correlated, it is clear from the visual evaluation that the biphasic simulation deviates much less from the experimental results.

Figure 4.10 Reaction force response in indentation experiment B: elastic and biphasic simulations versus reported experimental. (3206 particles)
Both the elastic and biphasic case undershoot their anticipated reaction force values with respect to the experimental. In the biphasic case, this can partially be attributed to using the same material parameters that were defined for indentation experiment A, while the Young’s modulus given by Vidal-Lesso et al. for the elastic simulation is higher and should not undershoot. Acknowledging this discrepancy, it was theorized that the total surface area of the cartilage may have an effect on the reaction forces. Vidal-Lesso et al. states that experiments were performed on the complete compartments of the specimens, which is not reasonably represented by a narrow plug. For this reason, new simulations were run of indentation experiment B with the cartilage instead represented by a box measuring 15 mm square and the same 3 mm depth, composed of 6696 particles. The results are presented in Figure 4.11. While the elastic simulation still undershoots the anticipated reaction forces, the biphasic simulation is now nearly precisely aligned with the experimental (with respective correlations of \( r=0.993 \) and \( r=0.996 \)). This shows that biphasic components are necessary not just for the approximation of relaxation phenomena and component stresses, but for accurate determination of cartilage behaviour as a whole.
Figure 4.11 Reaction force response in indentation experiment B with increased surface area: elastic and biphasic simulations versus reported experimental. (6696 particles)

The dependency on cartilage surface area can be explained by mechanical effects at the edges of the cartilage, but also in the biphasic case by the load bearing fluid mass. Particles at corners and edges of a sample have more surface area exposed to the external volume, which during compression will have a lower pressure than nearby particles, allowing fluid to flow outward more easily. This effect can clearly be seen when comparing the relaxation phases of the biphasic simulations in Figure 4.10 and Figure 4.11. Figure 4.11 exhibits a steeper negative slope but displays a higher peak reaction force, meaning that more fluid is contained in the deformed area immediately prior to relaxation.
Figure 4.12 Reaction force response in indentation experiment A with increased surface area: elastic and biphasic simulations versus reported experimental. (4774 Particles)

For completeness, it is necessary to return to indentation experiment A and similarly enlarge the surface area of the represented cartilage, with the results shown in Figure 4.12. While the previous results from indentation experiment A were good, simulating a greater surface area composed of 4474 SPH particles provides even better correlation, even for the elastic case ($r_{\text{elastic}} = 0.985$, $r_{\text{biphasic}} = 0.994$). Therefore, when performing such simulations it is important to be conscious of the cartilage surface area.
Figure 4.13 Visualization of stress distribution of indentation experiment B. A) Initial condition (0 s). B) Maximum strain (2.38 s), C) End of relaxation (3.88 s), D) End of simulation (6.26 s). (6696 particles)

Visually, indentation experiment B exhibits similar trends to the smaller particle visualization of indentation experiment A, with the increased particle number providing with an enhanced clarity. Specifically this relates to the transition from Figure 4.13B to Figure 4.13C, where relaxation due to fluid mass diffusion has clearly reduced the stress in the central area of the cartilage around the indenter and spread to the surrounding regions. Also it is much more evident that the points of highest stress can easily be
determined through visual examination, thereby fulfilling one of the requirements of the current research.

The pore pressure distribution shown in Figure 4.14A is more evenly distributed than the previous example in indentation experiment A, likely due to the larger surface area. Rather than examine this from a different angle, it is much more instructive to evaluate the pore pressure in the cartilage at the end of the simulation when the indenter has fully retracted from contact with the particle surface. This is shown in Figure 4.14B, where the area of negative pore stress corresponding to the swelling material can clearly be seen beneath the indenter. However, the distribution of poor pressure is very uneven.

The likely cause of this inconsistency becomes clear with a detailed examination of the indenter geometry. Figure 4.15A shows that the particles have been initialized by the Maya script so that there is a considerable offset to one side, and there is even an extra vertical column of particles in the top left corner. Figure 4.15B highlights a missing particle at the end of a single column of particles, while on the opposite side of the indenter there are instead three full columns. These irregularities result in deformations occurring sooner and spread over a larger area in the cartilage associated with the left side of Figure 4.15A.
Figure 4.14 Pore stress distribution of indenter experiment B (increased surface area): A) Maximum strain (2.38 s), B) End of simulation (6.26 s), noting the faster recovery in the area opposite the offset indenter particles.

Figure 4.15 Irregularities of the geometry of the indenter used in indenter experiment B due to method of Maya initialization: A) Particles offset to one side and extra vertical column of particles, B) Missing lower particle and single particle column compared to three columns on opposite side.
Referring back to Figure 4.14A, the pore pressure does not appear to have increased at the time of maximum strain at the edges of the cartilage furthest from the offset indenter particles, unlike those directly opposite the offset particles. Furthermore, Figure 4.14B shows that the area around the offset particles has a higher pore pressure at the end of the simulation while the opposite side has recovered more quickly (noting that there is currently no way to visualize negative pressure). This behaviour was verified from a complete evaluation of the entire process from beginning to end of deformation, which implies that greater deformation occurs in the area of offset particles. In addition, from visual evaluation of the fibril stress distribution in Figure 4.16, the offset area does indeed correspond to the offset particle area, causing a similar “bulge” to that noted in indentation experiment A.

**Figure 4.16** Fibril stress distribution of indenter experiment B (increased surface area) at 2.38 s: A) front view, B) opposite view, noting the increased fibril stress in the area of the offset indenter particles (left).

Clearly the irregular geometry of the particle configuration initialized by Maya to represent an indenter, which was based on a perfectly cylindrical 3 mm diameter mesh, is
not ideal. Nevertheless, the fluid in the cartilage appears to behave realistically considering the forces imposed upon it by the simulation. This may even be considered a more thorough test of the procedure that in it otherwise would have been, since despite the indenter irregularities, the simulation was still able to produce an accurate reaction force. After all, actual geometries within the acetabulum are not likely to be remotely symmetric.

4.2.3 Unconfined Compression Experiment

Indentation experiments A and B are very similar and although their results are encouraging, by themselves they do not provide suitable evidence to confirm the validity of the current procedure. For this reason, a third experiment was performed based on the unconfined compression research of Li et al. [93]. Once again, the material parameters remain the same as those used for the previous indentation experiments. Unlike in those experiments, the cartilage non-axial dimensions are specified, which is important for the determination of reaction forces since the compression plate will be applied to the entire upper surface of the model. The cartilage is a 3 mm diameter plug of 1 mm depth represented by 98 deformable particles with a 3 particle depth, while the upper plate performing the compression is composed of 352 boundary and virtual particles. It is not necessary to simulate a bottom plate in this experiment since the bottom layer is again fixed.

This experiment is dissimilar to the indentation experiments in more than just the scale, however, as it was created by Li et al. to examine the effects of stress relaxation, so it has a much longer time frame of 60 seconds. The compression plate is applied to
produce a compressive strain of 15% in the first 15 seconds and then allowed to relax for
the final 45 seconds. As previously stated, the current procedure is unable to represent
stress relaxation on this long of a time frame, but it does allow for evaluation of the
procedure in the simulation’s calculation of the peak reaction force.

The model’s small scale poses a question related to the implementation of the void
particles to allow for fluid to exude from the cartilage material. In the previous
simulations, deformable particles that became neighbours to the boundary particles
comprising indenters had their void particles removed, effectively simulating non-porous
indenters. For the current experiment, implementing void particles in this way would
eliminate almost all outflow of fluid mass, since all but the bottom layer of deformable
particles composing the cartilage are neighbours to boundary particles during
compression. Alternatively, void particles could be created as usual with the restriction
for neighbouring boundary particles removed, thereby allowing fluid mass to flow freely
through the compression plate as though it did not exist. In the following experiment,
both simulations were performed to determine what effect void particles have on the
results.

Examining Figure 4.17, all three simulations performed produce similar results to the
experimental prior to the relaxation phase. Because the strain rate is much slower than in
previous experiments, the additional time has allowed a higher amount of fluid to leave
the cartilage when peak compression is reached, so the biphasic cases exhibit must less
relaxation. The peak reaction force experienced in the biphasic simulation without void
particles is higher because more fluid mass is contained within the cartilage since there is
no external flow. It can be seen during the relaxation phase that the biphasic simulations
Figure 4.17 Unconfined compression 1%/s strain rate. Comparison of reaction force results for simulations with active biphasic properties without void particles (“No Void”), with void particles (“Void”), and basic elastic versus reported experimental.

are slowly approaching the stable equilibrium value of the elastic simulation, which is to be expected, however it should be noted that unlike the simulation that uses void particles, the biphasic simulation without void particles will always have an internal fluid pressure preventing it from decreasing to this value. Again, none of the simulations should be expected to reproduce the exact experimental results presented by Li et al. since they are not using parameters specific to that experiment. No statistical correlation tests were run for the compression case due to the inability to match relaxation phenomena at a long time scale. In addition, it was found that with a slower strain rate the simulations exhibited much less fluctuation, with the elastic simulation varying only
between 12.557 and 12.556 for the last 15 seconds of the simulation where strain was held constant. The fluid mass diffusion in the biphasic simulations was similarly stabilized, indicating that the indentation experiments were operating at the limits of stability of the simulations while the compression experiment is not, and that said stability is highly dependent on strain rate.

Inspecting the visualization of the biphasic void particle experiment in Figure 4.18, as expected the cartilage has no initial stress (A), reaches a maximum stress at 15 s (B), and then gradually relaxes (C). The important thing to note from the visualizations is the moderately uneven stress distribution between particles due to the small scale, even in comparison to the results obtained from the indentation experiments. Again, this does not appear to affect the accuracy of the obtained reaction forces to a large degree.

**Figure 4.18** Unconfined compression with biphasic (void). A) Initial configuration (0 s), B) Maximum strain (15 s), C) End of simulation (30 s).

Investigation of the component stresses in Figure 4.19 (again, purely for illustrative purposes rather than numerical accuracy) reflects the increased portion of stress borne by the non-fibrillar matrix at peak deformation in comparison to the indentation experiments.
due to the additional time for outflow of fluid mass. There is also a very low average fibril stress (1-2 kPa at maximum), appearing to indicate that the even distribution of load over the upper surface of the cartilage volume results in very little tension in the fibrils. This will be investigated further in the visualizations. Inconsistency is exhibited in the relationship of stress to the attained reaction force, in that the total material stress decreases by approximately 50% during relaxation while the reaction force decreases no more than 1/10th of that amount. This is again thought to be a result of the way the stress scalars are calculated, as explained for the indenter experiments, and emphasizes the limited use of such a comparison. There is also a small but mysterious drop in the pore stress at the 8 second mark that does not appear in any other simulation.

**Figure 4.19** Component stresses for 1%/s strain rate unconfined compression.
Figure 4.20 Component stresses for 1%/s strain rate unconfined compression with calculated total material stress.

If instead the stress is calculated from the total reaction force, as explained in Li et al. in Figure 2 of that work, “The stress was averaged over the cross-sectional area for transient responses”, then the result is Figure 4.20. Unlike Figure 4.19, this is the precise relationship that would be expected based on the reaction force results. More importantly, it fits the expected relationship for a 1%/s strain rate if compared to the total material stress curves shown in Figure 2 of the work by Li et al.. Figure 4.20 not only verifies the accuracy of the pore stress determined through the simulation, but because the total material stress exhibits the expected trend considering the relationships shown in Figure 2
from Li et al., while also matching the trends of the force reaction curve of Figure 4.17, it also validates the experiment as a whole. While this method of total material stress determination should work for cases of compression where the force is spread evenly across the cross-section, unfortunately it will not work for indentation experiments or real-world scenarios where impingements cause stress concentrations. This is an open question that will need to be resolved in the future.

**Figure 4.21** Component stresses for 15%/s strain rate unconfined compression with calculated total material stress.

The higher strain rate simulation of 15%/s illustrated in Figure 4.21 does not show the mysterious drop in pore pressure of the 1%/s simulation, leading to the assumption that it
was a one-time anomaly in the simulation, but more simulations will have to be performed to verify this assumption. The same issue exists in relating the simulated results of the total material stress to the attained reaction forces, so only the calculated stress is shown in Figure 4.21. The pore pressure supporting a higher percentage of the load at higher strain rates is clearly illustrated, a result that is verified by Li et al. [93]. The fibrillar stress again does not make any meaningful contribution to the total material stress.

![Figure 4.22 Matrix axial stresses of 15%/s strain rate unconfined compression divided into layers.](image)

In an attempt to explain the disparity between the stress values and the reaction forces attained from the simulation (as opposed to calculated stress values), Figure 4.22 was
produced, which separates the axial stresses experienced by the non-fibrillar matrix into three layers, which is a simple procedure since the cartilage is only three particles in depth. The middle layer appears to balance the stress between the positive stress of the bottom layer and the negative stress of the top layer. As well, the axial stresses in each layer exhibit only slight responses to the decreasing pore pressure, much less than can be seen in the total matrix stress in Figure 4.20. It is unclear whether this is a reasonable result since no other research could be found that performs a similar investigation, but it does suggest that the fluid is responsible for the majority of the relaxation stress response.

![Graph](image.png)

**Figure 4.23** Calculated total material stress versus strain for 15%/s and 1%/s strain rate unconfined compression simulations.

Examining the relationship between calculated stress and strain rate presented in Figure 4.23, a strain rate of 1%/s yields stress results that are precisely in the range expected from Fig. 2 of Li et al., while a rate of 15%/s yields only a slight increase in stress in comparison. Considering the results in that work, where a 15%/s strain rate
produces more than double the stress of 1%/s simulations and where the fluid supports more than 98% of the total load in the initial compression at strain rates greater than 5%/s, this reflects the reduced contribution to the total load by pore pressure due to the restrictions of the simulation. Figure 4.23 shows the expected increase in stress given those restrictions.

![Figure 4.24](image.png)

**Figure 4.24** Component stress distributions of 1%/s strain rate compression experiment at maximum strain: A) Fibril stress, B) Pore stress.

The visualizations of the component stresses for the 1%/s strain rate simulation are presented in Figure 4.24. The compression plate has been hidden for clarity. During compression the cartilage expands radially, causing tensile stress in fibres in the superficial and, to a lesser extent, the middle layer oriented in the radial direction. The stress magnitudes in Figure 4.24A are much lower on average than in the indentation
experiments. It is unclear whether this is an accurate result or if the small scale of the simulation caused the fibrils to under-represent their stress contribution. Clearly, the fibril stress distribution in the superficial layer is drastically uneven due to the same geometry problems exhibited by the Maya initializations in previous experiments. It may be that a depth of three particles is insufficient to accurately represent the fibril network in compression, but more compression experiments will have to be performed with differently-sized cartilage plugs to determine if this is true. In comparison, the pore stress shown in Figure 4.24A is more evenly distributed.

### 4.2.4 Cartilage Experiments Summary

Tests in Section 4.2 were performed on an Intel Core i5-3450 CPU with 8GB of RAM and an NVIDIA GeForce 660Ti with 2GB of onboard memory. The frame-rate results summarized in Table 4.2 are comparable to those found in the elasticity simulations of Section 4.1.

<table>
<thead>
<tr>
<th>MODEL</th>
<th>VERSION</th>
<th>PARTICLES</th>
<th>BOUNDARY</th>
<th>FRAMES/S</th>
<th>CORRELATION - r</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indentation A</td>
<td>Elastic</td>
<td>1524</td>
<td>246</td>
<td>98.871</td>
<td>0.976742</td>
</tr>
<tr>
<td></td>
<td>Elastic</td>
<td>4774</td>
<td>286</td>
<td>46.1606</td>
<td>0.98481</td>
</tr>
<tr>
<td></td>
<td>Biphasic</td>
<td>1524</td>
<td>246</td>
<td>69.2433</td>
<td>0.993154</td>
</tr>
<tr>
<td></td>
<td>Biphasic</td>
<td>4774</td>
<td>286</td>
<td>32.8984</td>
<td>0.994124</td>
</tr>
<tr>
<td>Indentation B</td>
<td>Elastic</td>
<td>3206</td>
<td>246</td>
<td>62.7095</td>
<td>0.98623</td>
</tr>
<tr>
<td></td>
<td>Elastic</td>
<td>6996</td>
<td>286</td>
<td>32.6161</td>
<td>0.992574</td>
</tr>
<tr>
<td></td>
<td>Biphasic</td>
<td>3206</td>
<td>246</td>
<td>44.7148</td>
<td>0.98132</td>
</tr>
<tr>
<td></td>
<td>Biphasic</td>
<td>6996</td>
<td>286</td>
<td>23.2458</td>
<td>0.995798</td>
</tr>
<tr>
<td>Compression</td>
<td>Elastic</td>
<td>98</td>
<td>352</td>
<td>172.403</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Biphasic</td>
<td>98</td>
<td>352</td>
<td>138.116</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2 Summary of results for all cartilage compression experiments.
Correlation tests were not performed between the experimental and the simulations the compression experiment because of the effect of relaxation due to the lengthy duration of the experiment. However, all other tests show a strong correlation between the simulation and the experimental results. The correlation of the elastic simulations for indentation experiment B are slightly higher than the 3206 particle biphasic simulation (implying the relationship is more linear) but a visual examination verifies that the biphasic simulation deviates much less from the experimental if scale is considered. As can be seen in figures Figure 4.10 and Figure 4.11, employing biphasic effects to cartilage with a larger surface area yields results more strongly correlated to the experimental with less deviation from the experimental in all cases.

![Graph](image)

**Figure 4.25** Comparing elastic versus biphasic results for frame rate versus number of particles for all cartilage indenter simulations.
Figure 4.25 shows a mean drop in frame rate from elastic to biphasic simulations of 29.03%, with a standard deviation of only 0.62. This is considered to be an acceptable decline in performance for the increased accuracy of the biphasic simulations. As such, the biphasic simulations are considered here to be superior to the perfectly elastic simulations for the representation of cartilage compression behaviour. The unconfined compression simulations were not included due to the high boundary/deformable particle ratio. Lower particle numbers would likely reduce the linearity of the particle number and frame rate relationship, as shown in Section 4.1.
Chapter 5  Conclusions and Future Work

5.1 Conclusions

The current procedure employs SPH to reproduce accurate numerical results from experimental research on soft tissue, and is the first that could be found among the available body of works to attempt this. It is also unique in its ability to determine accurate reaction forces to compression deformation applied by rigid bodies in SPH simulations of any kind of material. Although this capability has only been applied to experiments involving cartilage in the current research, it could conceivably be used for many other types of simulations involving rigid bodies, meaning that its potential applications extend well beyond that of cartilage alone.

The current procedure uses at its base design the framework proposed by Becker at al. [5], which was then extended to operate on NVidia’s CUDA parallel programming platform, thereby significantly increasing the computational speed of the simulations. While an Octree approach to the neighbour search algorithm employed by the current SPH procedure was initially suggested to be superior in terms of computational speed, this was found not to be the case when applied to parallel programming, at least in the format attempted for the current procedure.
The elastic models as implemented in the current procedure were validated for accurate rotation handling, which is essential should the current procedure be applied to the representation of a hip joint that has components that rotate during common activities, such as walking.

Various methods have been proposed to represent the different cartilage components. A neo-hookean matrix replaces the basic hookean - or linearly-elastic - constitutive model employed in the works by Becker et al. to represent the non-fibrillar matrix. A new method has been implemented to represent collagen fibre stress contribution. The methodology proposed by Lenaerts et al. to represent internal fluid mass diffusion and stress response has been significantly altered to more accurately simulate the behaviour of biphasic cartilage. Furthermore, the external fluid flow in that work has been replaced by a new method with unlimited external mass capacity, referred to here as void particles, that imposes much less computational burden than would simulating an equivalent number of fluid SPH particles. Each of these components were successfully validated in their intended applications as they were proposed through the experiments in Section 4.2. The accuracy requirements for the simulation as defined in Chapter 1 have been met through the use of the biphasic components (sub-millimeter accuracy, <10\% deviation from peak reaction force for known material parameters and less than an order of magnitude difference for non-sample specific material parameters). Incorporating these components was found to decrease frame rate by an average of only 29\%. The added accuracy from their inclusion is suggested here to be advantageous enough to deem that decrease in speed acceptable.
In practice, there is a wide variation between cartilage depending on the age, pathology and a host of other factors, so it should not be expected that a single group of material parameters should exactly describe every particular case. This does not account for the natural variation in material properties between cartilage samples, so some variation in results between the simulation predictions and the experimental results should be expected. However, it was shown here that a single set of parameters can be used for all cases without experimentally determining them and reasonably accurate results can be expected. This is of great benefit to a preoperative simulation, since such a simulation would be based on MRIs, and patient specific cartilage samples would not be available for testing until after such an operation is performed and the cartilage has been removed.

The addition of void particles removed the necessity of explicit fluid particle creation, significantly speeding up the simulation. An SPH fluid simulation based on the Navier-Stokes equations was thoroughly tested in the current framework to produce visually pleasing results, and can be incorporated into a future simulation if desired, but is not considered necessary for the current research.

Osmotic swelling effects were ignored in these simulations as it was assumed they would not have a substantial effect on accuracy for the purposes of this research.

5.2 Future Work

The stability of the system is less than could be desired, resulting in a very small time step that removes the likelihood of a real-time simulation from the realm of possibility in its current state. Real-time here is considered to be 24 fps if each frame represents 1/24 of
a second in time. While many of these simulations do achieve 100 fps and greater speed, the necessity of a very small time step means they are operating at much less than real time. One reason for this is the use of the very simple leap frog integration scheme. An always stable method should be investigated as a replacement, even if this means converting to an implicit system. Implicit integration may mean that each frame takes ten times as long to complete, for example, but it may also allow for an increase in the size of the time step by a factor of 100 or more, bringing the simulation much closer to the real-time ideal. In addition, applying deformation at too high a strain rate causes the simulation to become unstable. This problem could conceivably be rectified for all realistic loading scenarios by switching integration methods.

Many more particles would likely be required for a whole hip simulation than were used in the example cartilage experiments. If this number should exceed the current 60,000 particle limit, steps will need to be taken to increase this value. While it has previously been suggested that the data corresponding to each particle could be reduced and also that the data may be sent in partial “chunks” so as not to overflow the GPU onboard memory, another, perhaps more obvious option, would be to use several GPUs or use a GPU with a larger amount of onboard memory. This is certainly a possibility, but it should be remembered that the objective of the current procedure is that it should be applied to a preoperative simulation of a hip joint and should be able to run on a “common” desktop PC. Therefore, there should not be an unlimited budget used to meet this requirement, reflecting how the application will be used in a real world environment.

The fluid mass diffusion in the biphasic simulations was stabilized at the lower strain rate of the compression experiment, indicating that the long time unconfined compression
simulations were not operating at the limits of their stability. More long time unconfined compression simulations could be run to determine how close to the experimental results the simulations could achieve with the fluid mass bearing greater proportions of the load.

The exact number of SPH particles required will be dependent on the coarseness of the originating mesh and on the desired particle density, but an approximate value should nevertheless be determined before proceeding with a whole hip simulation. It may be necessary to obtain imagery from MR scanners of greater magnetic field strength than the average scanners currently used for clinical evaluation to attain a three-dimensional image dense enough to adequately represent the thin layer of cartilage. It should be noted that acceptable results were obtained with the unconfined compression experiment with a depth of only three particles, so that may mean less particles are necessary for such a simulation than would perhaps be expected.

Currently, colourization in the simulations is based upon the average of the magnitudes of the three principal stresses, but any of the values associated with an SPH particle could conceivably be used. Furthermore, it should be investigated whether this is the most reasonable method of representing the stress values. The Von Mises stresses were also calculated, and were found to be in a similar range, and it would be a simple matter to output those values instead. As an extension to the current program, a single #define in the Vars.h file could be used to switch between colorization output. As well, although using a single channel of the RGB (Red/Green/Blue) output for colourization to represent each of the stress components made it easy to differentiate between simulations, it would likely be of greater value to employ Matlab’s colormap, which would more clearly contrast stresses in the visualizations.
While visualizing through the Maya API is a convenient way of establishing the functionality of the system, future versions of the program should allow for interactivity and visualization as the program is being run. The program should be relatively easy to convert to use OpenGL as a method for visualization due to the compartmental coding structure of the C++ program created for the current procedure, which would allow for these extensions to be realized.

The current method of initializing particle positions according to the meshes created in Maya produces solids with irregular particle distributions, however this is more than partially due to the small scale of cartilage in simulations with 0.5 mm particle spacing. The fibril and pore stress distributions were shown to be highly susceptible to particle initialization geometry. It will need to be clarified how disruptive an effect this will have when applied to the model of a real hip joint, but it should be noted that a real hip joint will not have the symmetric geometries of the previous experiments.

Additional research should be investigated or performed that describes the stress-strain relationships of the three experiments to determine the correctness of the simulation stress results, since they were reduced from a matrix representation to a scalar in a way that is not directly relatable to the stress contributions from other components. Considering the force response results upon which the stresses are based appear to be very accurate, it would be precipitous to discard the simulated stress results in their entirety, and instead should be used to illustrate the general trends experienced by each component.

Additional compression experiments should be performed and validated against experimental results. With only a single cartilage compression study to compare against,
it is difficult to state conclusively that the current method will be accurate in all conceivable cases. If the problem of stability were to be solved, allowing for true biphasic material parameters to be used in simulations, this concern would be trivial to alleviate as there are a plethora of compression relaxation studies available to compare against. Unfortunately, short time frame cartilage compression studies appear to be extremely sparse. On the other hand, as mentioned by Vidal-Lesso et al. [123], short time frame cartilage compression studies are comparable to physiological loading scenarios, and so would presumably be more applicable to identifying causes of cartilage damage due to impingements.

With the exception of the calculated results from the compression experiment, using the stress results from the simulations is currently not recommended until the questions as to their validity can be satisfactorily resolved. However, the visualization of stresses can be still used, as the stress values associated with particular colourizations will not change between simulations. In fact, it may be possible to assign a non-physical scale to the colour values to help determine whether the highlighted stresses are a cause for concern during their evaluation. If with more research it is determined that the stresses relate in a linear fashion to experimental results, whether it is in a 1:1 ratio or otherwise, it may be a simple matter of multiplying the output stress to make an accurate determination as to what those peak stress values should be.

Despite the inherent questions that remain about the usability of the stress values determined from this system, it is suggested here that the validity of the current procedure has been validated to an extent to warrant the extension to a model of a real hip joint attained from MRIs.
List of References


W. Wilson, C. C. van Dokelaar, B. van Rietbergen, K. Ito, and R. Huiskes, “Stresses in the local collagen network of articular cartilage: a poroviscoelastic fibril-


volume fraction distribution function. Mathematics and Mechanics of Solids, ”


Related Publications
