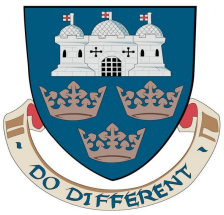
Simulating the Cardinal Movements of Childbirth Using Finite Element Analysis on the Graphics Processing Unit



Zelimkhan Gerikhanov

Supervisor: Dr. Rudy Lapeer

School of Computing Sciences

University of East Anglia

Degree of Doctor of Philosophy

March 2017

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my special appreciation and thanks to my supervisor Dr. Rudy Lapeer, who has been a tremendous mentor for me. It has been an honour to be his PhD student. I would like to thank him for guiding and encouraging my research and for allowing me to grow as a scientist. I thank my fellow lab-mates Dr. Vilius Audinis and Said-Magomed Sadulaev for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last four years.

Last but not the least, I would like to thank my family for supporting me spiritually throughout writing this thesis and my life in general.

Abstract

Many problems can occur during childbirth which may lead to instant or future morbidity and even mortality. Therefore the computer-based simulation of the mechanisms and biomechanics of human childbirth is becoming an increasingly important area of study, to avoid potential trauma to the baby and the mother throughout, and immediately following, the childbirth process. Computer-based numerical methods, such as the Finite Element Method, have become more widespread to simulate biological phenomena over the last two decades or so. One of the important aspects of such methods is them being able to accurately model the underlying physics and biomechanics of biological processes. In the case of the childbirth process, an important role is played by the fetal head and its motion as it is being born. The most important manifestations of the head's motion are described as the cardinal movements. Being able to model the cardinal movements in a computer-based model of the human childbirth process is compulsory as they occur in almost every normal delivery. Many existing simulations use reverse-engineered approaches to model the cardinal movements by imposing pre-defined trajectories that approximate a real childbirth. These approaches lack physical accuracy and are unable to extend the simulation to unseen scenarios where for example the childbirth process does not develop normally. To create a simulation software capable of simulating realistic, including unseen, scenarios, and which does not make any pre-simulation assumptions about the cardinal movements, a physical and forward-engineered approach in which the motions of the head are determined by the underlying physics, is required.

This thesis presents a simulation system where the physical behaviour of the fetal head is modelled during the second stage of childbirth. Accurate models of the fetal head and trunk, the maternal uterine cervix, bony pelvis and pelvic floor muscles, were created. Some of these are considered to be rigid bodies in the simulation (fetal head, trunk and maternal bony pelvis), whereas others are considered to be deformable (maternal uterine cervix and pelvic floor muscles. The Finite Element Method (FEM) is used to model the deformable tissues by implementing the Total Lagrangian Explicit Dynamics (TLED) approach on the GPU. The combined TLED/contact mechanics method was first tested on simple

hyperelastic objects. Following successful validation, the interaction between the fetal head and the deformable tissues was evaluated using a projection based contact method in conjunction with TLED.

Several experiments had to be done to find the required set of factors contributing to the occurrence of the cardinal movements. These steps are reported in the thesis as well as the results of the final experiments which do exhibit the key cardinal movements of a normal childbirth process, marking the successful, and key, contribution of this thesis.

The GPU based acceleration allows running the simulation in near real-time, which makes it possible to create interactive simulations for training purposes of trainee obstetricians and midwives. The simulation system presented in this work is also the first step towards a fully patient specific system that would allow clinicians to diagnose and/or predict adverse scenarios of childbirth based on the MRI scans of real pregnancies prior to labour.

Chapter 1

Introduction

1.1 Background and Motivation

Computer based simulations find numerous applications in medicine. Such applications include training of medical personnel, diagnosing patients based on digital data and scientific research to gain better understanding of physiological phenomena. Biomechanical simulations are one of the most challenging uses of computer based simulation in medicine. The underlying principles are very complex and thus require sophisticated theoretical formulations and considerable software development efforts.

The process of human childbirth is sometimes labelled as the most critical time of someone's life. Many problems can occur during childbirth which may lead to instant or future morbidity and even mortality. Therefore, the computer-based simulation of the mechanisms and biomechanics of human childbirth is becoming an increasingly important area of study, to avoid potential trauma to the baby and the mother during the process. The combination of all the phenomena and processes involved in childbirth comprize an intricately complex system. It is difficult ethically and practically to observe the system throughout the birth process. Childbirth may be a dangerous event for the child and the mother. Monitoring the event via interventional devices may introduce additional risks which is undesirable. Additionally, there are issues of legal and ethical approval, which can be very difficult to overcome. These and many other factors make it

difficult to study the childbirth phenomenon with great detail and accuracy. This leads to the fact that the study of childbirth and the computational applications therein can be seen as a niche area of science. The limited information available studied had to be studied in depth to maximise the potential of the proposed research.

One of the important aspects of such methods is being able to accurately model the underlying physics and biomechanics of biological processes. In the case of the childbirth process, an important role is played by the fetal head and its motion as it is being born. The most important manifestations of the head's motion are described as the cardinal movements. Being able to model the cardinal movements in a computer-based model of the human childbirth process is compulsory as they occur in almost every normal delivery.

The main focus of this project, which is the main deliverable of this thesis, is to create a realistic real-time computer based simulation of human childbirth with the ability to simulate the cardinal movements of human labour (see section1.1.1). Simulations of this kind require modelling biomechanical interactions of high complexity. As mentioned before, this implies sophisticated efforts to solve all the separate computational and design problems and then integrate the solutions in a coherent manner to arrive at an adequate simulation system capable of simulating childbirth in a forward-engineered manner. The main objective of this thesis is to give sufficiently detailed information about all the steps taken when implementing such a simulation system. The information should also be sufficiently complete to allow other interested parties to replicate the results and continue development of the system.

1.1.1 Importance of Cardinal movements

Many existing simulations use reverse-engineered approaches to model the cardinal movements by imposing pre-defined trajectories that approximate a real childbirth. These approaches are unable to extend the simulation to unseen scenarios where for example the childbirth process does not develop normally. To create a simulation software capable of simulating realistic, including unseen scenarios, should not make any assumptions about the cardinal movements, but

should follow a fully physical and forward-engineered approach in which the motions of the head are determined by the underlying physics and biomechanics of the process.

There is a consensus in the midwifery and obstetric profession stating that the fetal position and orientation while progressing through labour has considerable influence on the actual process of labour and outcomes for both the mother and the child (Fraser and Cooper, 2009). The cardinal movements and their realistic modelling will be an important part of the future predictive childbirth simulation system. A predictive simulation must be able to simulate the cardinal movements in a precise way, as they are a crucial factor in childbirth. Therefore, the main aim of this thesis is to describe the steps taken in order to achieve a physically realistic simulation of cardinal movements. All the other aspects of a childbirth simulation, such as patient-specificity, fetal and/or maternal injury estimation, intervention advice, etc, are left for future developments.

When the irregular form of the birth canal and the relatively large dimensions of a grown up fetal head are taken into consideration, it can be seen how some of the diameters of the fetal head cannot pass through the pelvis. This then suggests that for the birth to take place, the head needs to undergo a certain number of adaptation in order to accommodate the passage of the different dimensions of the fetal head through the different dimensions of the birth canal. The combination of the birth canal shape and the relatively large fetal head dictate that the passage of the head must be through the trajectory with minimal mechanical resistance. These modifications in the position of the presenting part constitute the mechanism of labor (Borell and Fernstrom, 1958).

A great amount of effort and extensive research has been done, but the knowledge of parturition remains scant. There have been numerious investigations of "the timing of birth" (Buhimschi et al., 2003), that is the kinematics of the process. However, there is still a limited understanding of the biological mechanisms that control the mechanisms of delivery. The methods investigating the ways to prevent these mechanisms from acting inappropriately are also limited. A multitude of factors and structures are involved and only partial understanding is currently available. For example the optimal contractile forces that lead to successful delivery are known, but the way of achieving these and the mechanics that

make them optimal are not known (Buhimschi et al., 2003).

The effects of childbirth on the female pelvic floor organs is of particularly great concern. The female post-partum pelvic organ prolapse is being studied (Petros, 2011), but further investigations need to be undertaken. The computer based simulations present a convenient tool for the study of the effects of childbirth on organ prolapse. The majority of the works presented. It is emphasised in this thesis that the effects of the childbirth on the movements of the fetal head is of main focus. This does not, however, preclude this study from being irrelevant for the investigation of causes and the solutions of the pelvic organ prolapse. As it will be demonstrated, the purely physical simulation of childbirth without imposed trajectories has direct benefit on the study of the integrity of the pelvic floor, as one of the most important factors in prolapse. Such type of childbirth simulation allows higher fidelity and thus leads to more accurate results of the experiment.

Thus it can be concluded that the cardinal movements play an essential role in allowing the head to successfully navigate the birth canal. Therefore, the following sections will be all dedicated to the anatomy, physiology and mechanics of the parturition. The methods of computer simulation allowing to model all of these will also be presented.

1.1.2 Problem statement

Great benefits can be achieved with further advancements in the theory and the technology required for high fidelity computer simulations. One of such achievements is the creation of a patient specific system capable of high-fidelity simulations of childbirth outcomes. The system should be able to predict the most probable outcomes of a given childbirth based on various CT or MRI scans and other patient specific data, such as the maternal age, medical history, etc. All this data could be combined to comprise a complete FE model and a simulation could be run. When a multitude of simulations are run in parallel with small variations of the initial parameters, the combined results of them could be used as an indicator of the likely outcome of the given childbirth before labour starts. This information can be useful for planning interventions, which carry much less risk

than an emergency intervention if the complication in labour was not foreseen.

At this time we are at the stage when such a system cannot be created without essential preliminary steps. One of the most important aspects of childbirth simulation is concerned with the mechanics of the process. When considering the mechanics of human parturition it is essential to study the motions and interactions of the maternal and fetal body parts involved in the process. Therefore, the research goal of this thesis is to have a simulation system capable of successfully and consistently producing the cardinal movements of human childbirth (listed in Section 1.4) in a fully forwards engineered by applying the FEA.

1.2 Anatomy

1.2.1 Maternal anatomy

One of the most important roles in childbirth is played by the maternal organs of the pelvic cavity and the bony pelvis itself. The main substructures of interest are (Figure 1.1):

- bony pelvis
- levator ani muscle complex
- uterine cervix
- sacrospinous ligament
- tendinous arch

The image in Fig. 1.4 demonstrates the various types of female pelvic shapes. The classification was introduced by Caldwell and Moloy (1938). The most important distinction between the different types is contained in the shape of the pelvic inlet. The variation between the different types of pelvises is also based on the relationship between the size of the pelvic inlet and outlet.

- Gynecoid – pelvis with a round inlet and spacious pelvic cavity. The overall height of the pelvis is smaller than a male's pelvis and other types of female

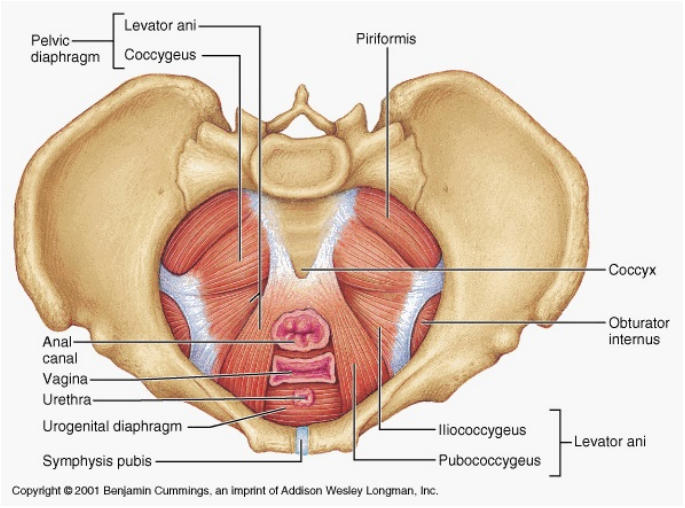


Figure 1.1: Superior view of the material pelvic area with indications of important substructures.

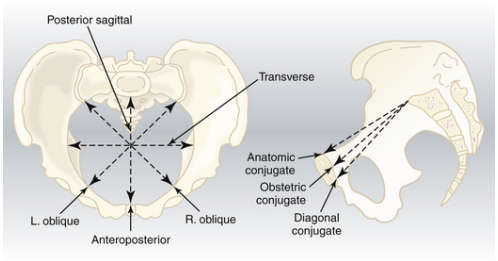


Figure 1.2: The main obstetric diameters of the pelvic inlet.

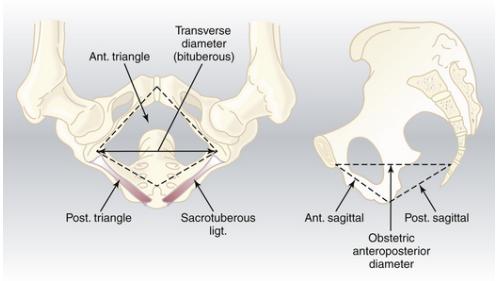


Figure 1.3: The main obstetric diameters of the pelvic outlet.

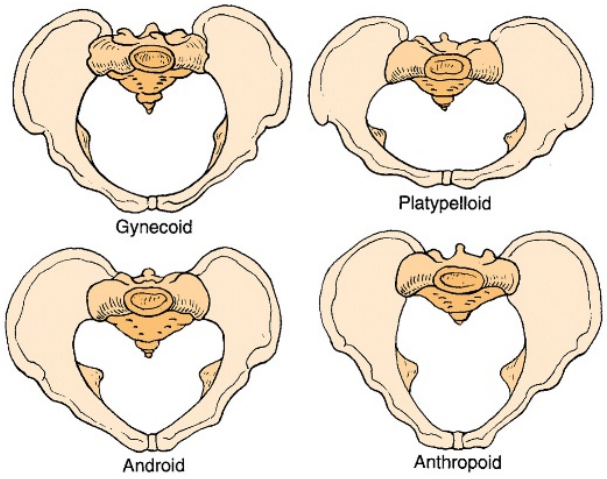


Figure 1.4: The different types of the female bony pelvis demonstrated. The types are differentiated based on the shape of the pelvic inlet.

pelvises. This type of pelvis is considered to be the most beneficial during pregnancy and during labour as its shape permits the fetal head to be accommodated better.

- Android – pelvis with a "wedge" like shape of the inlet. The inlet is wider in the posterior region and narrows down when approaching the pubic symphysis. This type of pelvis is more widely found in males and is not particularly facilitating childbirth.

- Platypelloid – is a pelvis with an oval shape with the antero-posterior diameter being considerably narrower than the transverse diameter.

- Anthropoid – pelvis with a wider antero-posterior diameter when compared to the transverse diameter as opposed to the Platypelloid type.

The visual subdivision of the pelvic floor is shown in Fig. 1.6. It can be seen that the subdivision follows a pattern of radial bands receding from the central area. The pelvic floor itself is a complex muscle structure consisting of a number of substructures which are listed in the order adhering to the receding radial band pattern:

- puborectalis

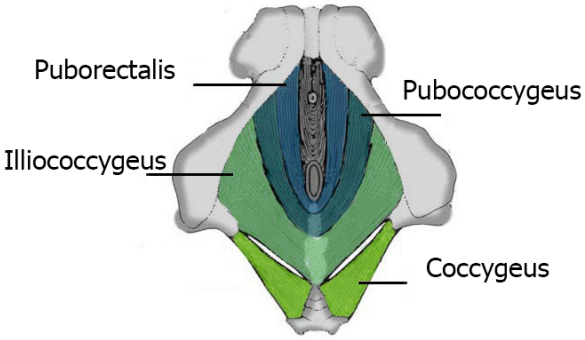


Figure 1.5: The substructures of the maternal levator-ani muscle as seen from the inferior view.

- pubococcygeus
- illiococcygeus
- occygeus

The tendinous arches are connecting the internal side of the pubis to the ischinal spines. They are also attached to the upper rim of the pelvic floor overlapping the pubococcygeus and illiococcygeus.

The ligament connecting the coccyx area of the sacrum and the iscihnal spines are known as the sacrospinous ligament. The ligament connecting the sacrum with the pelvic tuberocities is known as the sacrotuberous ligament. These two ligaments are closely connected to the substructures of the pelvic floor and may play an important role in the progression of labour and the cardinal movements in particular.

1.2.2 Fetal anatomy

Fetal head diameters

The anatomy of the fetus is another important subject of study when trying to investigate the causes of the cardinal movements.

The average circumference of the term fetal head, measured in the occipitofrontal plane, is 34.5 cm. The most important diameters of the fetal head are listed below (Hacker et al., 2015).

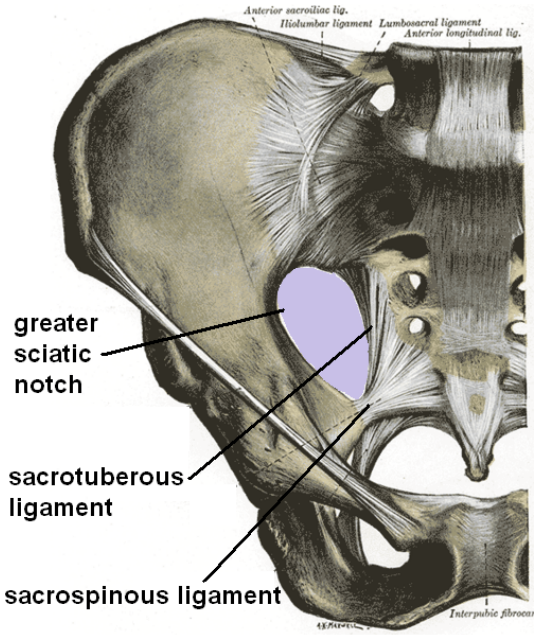


Figure 1.6: The sacrospinous and the scacrotuberous ligaments demonstrated. Image adapted from Häggström (2014)

1. Suboccipitobregmatic diameter – SOBD (approx. 9.5 cm), the presenting anteroposterior diameter when the head is well flexed, as in an occipitotransverse or occipitoanterior position. The diameter extends from the undersurface of the occipital bone at the junction with the neck to the center of the anterior fontanelle.

2. Occipitofrontal diameter - OFD (approx. 11 cm), the presenting anteroposterior diameter when the head is deflexed, as in an occipitoposterior presentation. The diameter extends from the external occipital protuberance to the glabella (lower forehead).

3. Supraoccipitomental diameter SOMD (approx. 13.5 cm), the presenting anteroposterior diameter in a brow presentation and the longest anteroposterior diameter of the head; it extends from the vertex to the chin.

4. Submentobregmatic diameter (SBD) (approx. 9.5 cm), the presenting anteroposterior diameter in face presentations. Extends from the junction of the neck and lower jaw to the center of the anterior fontanelle.

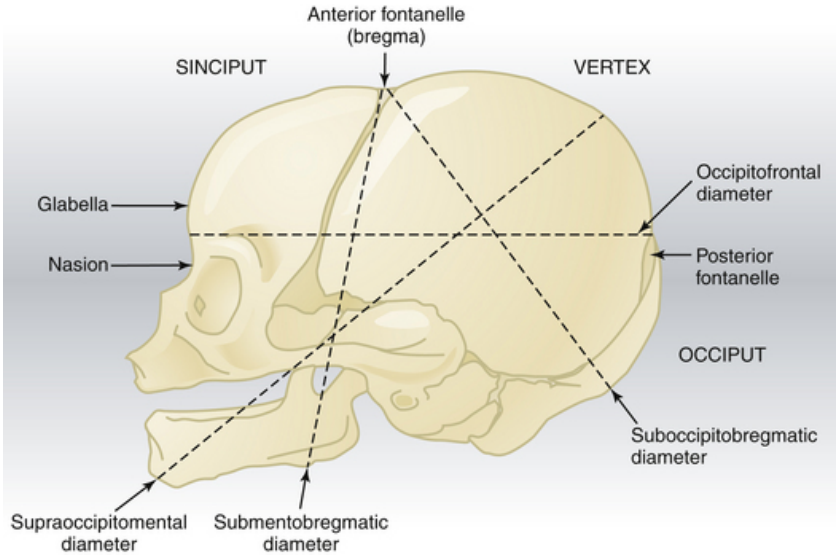


Figure 1.7: Fetal head diameters. (Hacker et al., 2015)

.

5. Biparietal diameter – BPD (approx. 9.5 cm), the largest transverse diameter. Extends between the parietal bones.

6. Bitemporal diameter - BTD (approx. 8 cm), the shortest transverse diameter. Extends between the temporal bones.

1.3 Human Childbirth

When considered as a whole, the process of human childbirth is a relatively long process. The gestation period is considered to be normally 40 weeks long. However, it often varies from the average by 3 weeks most often falling between 37 and 43 weeks. There are three stages:

1. The first stage is represented by slow descent of the fetal head towards the pelvic inlet until the engagement of the pelvic rim. It is also important to note that the first stage also involves the process of effacement, whereby the fetal head engages the uterine cervix causing it to dilate slowly, as this process may affect the next stage.

2. The second stage is represented by the traversal of the birth canal by the fetal head. This stage is when the cardinal movements take place.

3. The third stage is initiated when the baby is fully born and concluded by the birth of the placenta.

The thesis is focused on the second stage. In the case of natural labour the traversal of the birth canal by the fetal head occurs through a set of mechanisms known as *mechanisms of labour*. Sometimes they are also called the *cardinal movements*. A detailed account on the cardinal movements (which were introduced in section 1.1.1 is given in the following section.

1.3.1 Second stage

Normal vertex presentation

Adverse presentations

There are several ways in which the fetal head can present during the second stage of labour. The various ways it can present are classified as follows:

1. suboccipitobregmatic or occiput anterior presentation (OAP) presentation (Fig. 1.8 A). The head is flexed and the occiput is the presenting part.

2. occipitofrontal presentation or the transverse presentation (Fig. 1.8 B). The head is not flexed and the tip of the head (anterior fontanelle) is the presenting part.

3. mentovertical presentation or the brow presentation (Fig. 1.8 C). The head is partially extended and the fetal forehead is the presenting part.

4. submentobregmatic presentation or the face presentation (Fig. 1.8 D). The head is completely extended and the fetal face is the presenting part.

Fig. 1.8 shows the above head presentations. The occiput anterior presentation (OAP) seen in the figure is considered to be the normal and most common presentation provides the smallest diameter of the fetal head to be engaged with the pelvic inlet and outlet.

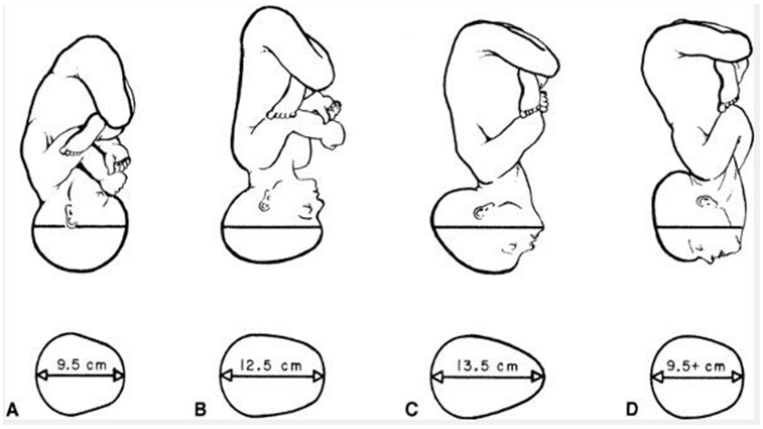


Figure 1.8: Fetal head presentations: a – suboccipitobregmatic (nornal) presentation, b – occipitofrontal presentation, c – mentovertical (brow) presentation, d – (submentobregmatic) face presentation. Adapted from Gabbe et al. (1991).

1.4 Cardinal Movements

The cardinal movements are only manifested during the second stage of labour described above. They are of great importance for this thesis and should be considered separately with additional detail.

The fetal head undergoes a set movements during the second stage of labour. The movements are descent, engagement, internal rotation, extension, external rotation, restitution and expulsion. These are referred to as "mechanisms of labour" or "cardinal movements".

Investigating the mechanisms of labour is an important preliminary step before designing childbirth simulation software. The process of childbirth is complex and involves a multitude of different mechanical processes. Gabbe et al. (1991) describes the cardinal movements as the main mechanisms of labour and lists seven distinct movements: engagement, descent, flexion, internal rotation, extension, external rotation (restitution) and expulsion. Liao et al. (2005) gives the following definitions to the movements and provides figures:

1. Engagement – this step is identified by the passage of the widest diameter of the fetal skull through the pelvic inlet (Fig 1.9). The widest diameter is the Fronto-occipital diameter and the widest pelvic inlet diameter is transverse diameter. Therefore, the head will engage while facing along the maternal

Figure 1.9: Engagement

Figure 1.10: Descent and flexion

Figure 1.11: Internal rotation

Figure 1.12: Extension

transverse axis as seen in Fig. 1.9

2. Descent – the stage when the fetal head passes further downwards through the cervix towards the birth canal (Fig 1.12).

3. Flexion – is the process when the fetal head flexes with its chin approaching its chest. The shape of the pelvis and the pelvic floor are said to cause flexion. Flexion allows the fetus to pass through the birth canal with a smaller diameter (e.g. the bi-parietal diameter (Fig 1.7) (Fig 1.10).

4. Internal rotation – is also thought to be caused by the shape of the pelvic canal and soft tissues. It is represented by rotation of the fetal head from facing sideways to facing backwards relative to the mother's body. This can be explained by the fact that the pelvic inlet has the widest diameter in the sideways direction, whereas the pelvic outlet is widest in the sagittal axis (Fig 1.11).

Figure 1.13: External rotation

Figure 1.14: Expulsion

5. Extension - happens while the neck of the baby is under the pubic symphysis. During extension the head deflexes and in this stage, chin and head are born facing outwards (Fig 1.12).

6. External rotation (restitution) is represented by external rotation of the head to face sideways as compared to the rest of the body (Fig 1.13).

7. Expulsion - when external rotation is complete and the anterior shoulder has moved under the pubic symphysis the fetus is born (Fig 1.14).

Fetal head trajectory

Known as Curve of Carus. One of the most important notions in the thesis is the trajectory taken by the fetal head while progressing through the maternal birth canal. The trajectory is not only represented by the translation motion of the fetal head's center, but also includes the rotations.

1.4.1 Childbirth simulators

There are a number of mechanical childbirth simulators that exist already. Such simulators are designed to allow trainee obstetricians and midwives to interact with a mannequin of a birthing woman. The mannequin is normally made of plastic and/or metal.

The motivation behind using a computer based simulation in childbirth modelling is dictated by a number of reasons. While having certain advantages the

mechanical childbirth simulators often lack very important features. Primarily, mechanical mannequin will typically be very poorly customizable. Such simulator is typically used for training junior personnel and in many of training cases the simulation scenario is required to be changed based on the type of the case that is being practised. Using mechanical simulator means that only a specific scenario is available for training with only slight variations. Additional mannequins of a different type will have to be acquired to perform training for alternative scenarios. Contrary to this, computer based simulations allow unrestricted customization. It is also worth mentioning, that there exists a hybrid type of simulators, that combine a computer based underlying bio-mechanical model with an external mechanical mannequin to provide the interface between the trainee and the simulator. Such, simulators combine the advantages of both types of simulators, but also carry the drawbacks of at least one of them.

Computer simulations provide a useful tool for representing real world phenomena, but they are only capable of representing the simulated objects to a certain degree of approximation. Better approximations are predominantly much more expensive in terms of computational power required for the simulation. With the increased processing power of modern computers, it is possible to perform simulations with a higher degree of fidelity. However, even the most high-performance machines can struggle with certain types of high-cost simulations. In such cases, we have to utilize the underlying hardware to the highest degree possible. This can be achieved by a number optimization techniques. One of the most effective techniques is using parallel processing in order to speed up the computation. Therefore a part of this thesis will be dedicated to describing the various parallelization techniques utilized in the simulation system.

1.4.2 Reverse vs Forward engineered approaches

There are a number of existing mechanical and computer based simulators of human childbirth. It is crucially important to contrast the approach chosen for this research project from the existing reverse-engineered simulations.

One of the most important contributions of this thesis is that a fully forwards engineered simulation of childbirth is produced as its part. The forwards

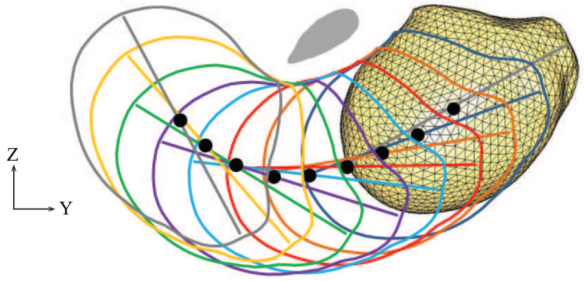


Figure 1.15: An example of an imposed trajectory applied to a fetal head. Image from Dejun Jing, James A. Ashton-Miller (2013).

engineered approach is opposite the reverse engineered solutions. In the case of the forwards engineered simulation, the simulation fully relies on the physical models to simulate the phenomena that occur during the labour. As opposed to this, the reverse-engineered solutions will use predefined animations to "fake" the phenomena.

A review of the current state of the art work in childbirth simulators is provided in Section 2. The review shows that many of the papers listed used reverse engineering where the fetal head movement is concerned. This renders these simulations at the very least incomplete and in some potentially physically inaccurate. In fact it appears that many computer based childbirth simulators ignore the mechanics of cardinal movements altogether, which strongly suggests lack of physical fidelity when compared to the real life phenomenon. Some reverse-engineered simulators rely on animated sequences of the cardinal movements which means their simulation component of the mechanisms of labour is non-existent.

The above mentioned reverse-engineered simulations are using superimposed trajectories on the fetal head. The fetus or only the fetal head is forced to follow a required trajectory without considering the causes of these imposed motions. Even with reverse-engineered trajectories the cardinal movements are in most cases not rendered correctly.

To give an illustrative example of how imposed trajectories are used in such simulations a figure 1.15 from work by Dejun Jing, James A. Ashton-Miller (2013).

1.4.3 Causes

The series of motions undertaken by the fetal head are collectively known as the cardinal movements of labour. Generally speaking all of the cardinal movements are caused by the birth forces and the interaction of the fetal structures with the maternal structures in the vicinity of the fetus as it progresses through the birth canal. The forces occurring as the results of the interactions between the structures involved are not studied in detail. Even though there is good general understanding of the process in terms of the stages involved (Stephenson and O'Connor, 2000; Hacker et al., 2015), the particular contributions of each of these interactions has not been investigated in detail.

Internal rotation One of the earliest investigations of the cardinal movements of labour were undertaken by Paramore (1909). The author provides an exhaustive and critical overview of the existing at that time ideas on the causes of the internal rotation.

Paramore (1909) follows arguments concerning the importance of the shape of the ischial spines. It is argued that the inwards projection of the ischial spines is not a "specially" evolved feature of the human skeleton to facilitate the occurrence of the internal rotation. The internal rotation is essential for the delivery of the fetal head and the inward projection of ischial spines is essential for the internal rotation to occur. The argument by Paramore (1909) is that the ischial spines are evolved as part of the change towards an erect posture, but have no connection to the internal rotation directly. That is they did not evolve to cause it, but rather the internal rotation is caused by the fact that ischial spines are present. It is argued that the causality is inverted and its the internal rotation that is caused by the presence of the inwards pointing ischial spines, but their absence would not prevent the head from being born without the internal rotation occurring. The main relevant conclusion is that the internal rotation is caused by the contact with the ischial spines.

Some of the earliest observations of these movements were done using various radiological instruments by Borell and Fernstrom (1957, 1958); Borell and Fernström (1959). The prior works would only qualitatively describe the movements

of the head based on the position of the head when the birth has already occurred (Paramore, 1909). Alternatively, the experiments would be done on post-mortem cases where the mother and baby would not survive the labour (Edgar, 1916). The radiological study was the first to allow observing the labour and the movements of the fetus as the labour progresses within a live subject. The results of the radiological study were used to formulate a detailed list of the movements that comprise the mechanisms of labour. These results were used as basis for the established theory of cardinal movements used in the contemporary obstetrics.

Further work by Borell and Fernstrom (1958) investigates the mechanisms of fetal shoulder rotation in the later stages of labour. The terminology of internal rotation is also attributed to the rotation of the shoulders as well as the head and in both cases refers to the rotation occurring when the head or solders engage the pelvic floor.

Sellheim (1924) suggested that the internal rotation is the effect of the interaction between the fetal body (as a whole) with the maternal pelvic canal. The bent nature of the canal causes the fetal body to bend to adapt its shape to the canal. However, the flexibility of the fetus is different around different axes. The most flexible axis is said to be the transverse axis, so that the fetus is most flexible when flexing forward and extending backwards. This asymmtery leads to the fetus orienting itself to go through the bent birth canal while rotating around the transverse axis. For this to occurr the fetus needs to rotate internally. This idea is strongly supported by Rydberg (1935).

Borell and Fernström (1959) propose an explanation for the occurrence of the internal rotation. Whereby the rotation is attributed to the complex interaction of the fetal head and maternal pelvic shapes along with the interaction of the fetal head with the maternal pelvic floor. This view is disagreement with the theory of Sellheim (1924) and the radiological studies of Borell and Fernstrom (1957), Borell and Fernstrom (1958) and Borell and Fernström (1959) appear more reliable.

Some modern medical literature also states that the shape of the bony pelvis causes the internal rotation. For instance, Henry et al. (2011), when describing the cardinal movements, only mentions the contact with the pelvis as the force rotating the fetal head.

The primary cause of the cardinal movements has also been attributed to the irregular form of the birth canal and the size of the fetal head at full gestation. This implies that not only the bony structures may have considerable effect on the mechanisms of labour. The head must adapt its orientation. One of the earlier works on this principle is by Geiger (1993) where it is shown that the head successfully negotiates through the birth canal when some cardinal movements are observed, but the physics model used in this study was simplistic. Similar qualitative analysis is provided by Martins et al. (2007) where they investigate the effects of cardinal movements on the pelvic floor. The limitation here is that the main focus is on the effects on the pelvic floor rather than effects of the pelvic floor and other structures on the fetal head.

Varnier (1900) showed that the bony pelvis of itself was insufficient to cause the cardinal movements to be observed. A series of experiments were undertaken where a fetus was pushed through the birth canal of a bony pelvis, which was cleaned from all soft tissues. They observed that no cardinal movements occurred during the experiment, which suggests that the interaction between the bony maternal pelvis and the fetal head does not cause the cardinal movements.

At the same time there is an opinion that it is primarily the interaction of the fetal head with the pelvic floor that cases the cardinal movements. This view is attributed to Hart and often referred to as the Hart's approach (Hart, 1912). The reasoning is that the larger occipital area of the fetus is able to progress further in the direction of the vaginal opening. It is claimed that the lesser resistance from the pelvic floor is in the anterior region closer to the vaginal opening. This generates a rotational torque forcing the head to rotate internally, with the occiput moving anteriorly.

Edgar (1916) undertook a series of experiments on a post-mortem case. The fetal head was propelled through the maternal birth canal manually. The results showed that the head internally rotated. The rotation was observed several times, but later the rotation had stopped due stiffening due to rigor mortis.

The findins of Varnier (1900), Edgar (1916) and the approach of Hart (1912) suggest the importance of the soft tissues in the occurrence of the cardinal movements.

There are even opinions which state that the fetal trunk rotates before the

rotation of the fetal head occurs. It states that the internal rotation is caused by the pulsating uterine force. Eichstedt (1859) proposed that the shape of the uterus changes considerably. The change in the uterine change then affects the orientation of the fetal body, which in turn forces the fetal head to follow by internally rotating. Olshausen (1906) had a similar opinion about the causes of the internal rotation, but additionally also considered the shape of the pelvic floor as another important contributor to the internal rotation of the fetal head.

Extension There have been two different theories proposed, when considering the extension. Sellheim (1924) claimed that extension of the fetal head takes place after impingement of the suboccipital region of the fetal head at the mother's pubic bone. A contact with the transverse axis running through the lower border of the pubic bone was thought to be the cause of the deflexion of the fetal head at the base of the skull and extension of the fetal neck. The main idea is that source of the rotation is at the maternal pubis.

A set of radiological investigations of the extension suggested that the theory of Sellheim (1924) where the rotation is caused by the interaction of the head with the maternal pubis was incorrect (Borell and Fernstrom, 1957). The head was seen continuously descending past the maternal pubis. The fetal spine undergoes extension only during the last stage of labor. A relatively large distance between the head and the lower border of the pubis was observed. Similar observations were made by Bamberg et al. (2012) using dynamic MRI.

1.5 Thesis Overview

1.5.1 Contributions

Real-time interactive FE simulation using GPU acceleration

A finite element method (FEM) based simulation of soft tissues was implemented (Section 3.2. The implementation is accelerated by the use of the graphical processing unit (GPU) to achieve real-time simulation rates (Section 4.2.

Mechanical model for efficient soft vs rigid tissues contact

The contact between the the soft maternal tissues and the bony skull is a persistent phenomenon in childbirth simulation. The contact must be efficiently detected and resolved. A contact method suitable for these purposes is presented and evaluated (Section 3.3). The accuracy of the method and its robustness are also presented (Section 5.2).

Fully physics based childbirth simulation system

The GPU implementation of FEM and the proposed contact method were utilized to achieve a full physics based simulation of childbirth (Section 5.5). The implementation details of the implemented system are outlined (Section 4.2). As opposed to reverse-engineered solutions, where the motions of the head are artificially imposed, our simulation features a realistic physical behaviour.

Experimental study of the cardinal movements

The fully physics based simulation of childbirth was used to investigate the mechanics of the cardinal movements. The physical mechanisms of the occurring head motions have been visualized and reported (Section 5.5). The effects of varying several simulation parameters have been studied and reported.

1.5.2 List of publications

In (Gerikhanov et al., 2013) we investigate the theory according to which the cardinal movements are caused by the contact between the bony maternal pelvis and the fetal skull. The results showed the theory to be incorrect.

In (Lapeer et al., 2014a) we investigate the effects of symmetric and asymmetric placement of the obstetric forceps on the fetal head. The results indicate that the asymmetric placement can result in major increase in the stresses occurring in the fetal skull. The author of this thesis was involved in developing the tools required for the experimental setup and also running the actual experiments in Abaqus CAE [1]. The results reported in the paper can be useful for the

[1] https://www.3ds.com/products-services/simulia/products/abaqus/

simulation system described in this thesis when considering obstetrical training scenarios. The results became part of the dissertation by Audinis (2017). The trainees' forceps placement skills could be assessed based on the results of the paper (Lapeer et al., 2014a).

In (Lapeer et al., 2014b) we investigate the effects of the correct and incorrect placement of the obstetric vacuum extraction tool. The incorrect placement, whereby the suction cup overlaps the anterior fontanelle was shown to have major effect on the fontanelle that could lead to adverse effects. The author of this thesis was involved in developing the tools required for the experimental setup and also running the actual experiments in Abaqus CAE. Similar to the results presented in (Lapeer et al., 2014a), the results of this paper can be used when training to execute vacuum extraction scenarios.

Chapter 2

Literature review

2.1 Introduction

The theoretical background required for creating a childbirth simulation covers a wide range of subject areas. These include both technical and clinical subjects related to the childbirth process. This review chapter aims to cover some of the most important scientific research in the relevant areas.

The first topics that are covered are related to the anatomy and biomechanics of the fetal head as it plays an important role in the phenomenon of the cardinal movements. Next we discuss the maternal pelvis anatomy and material properties. This is followed by a detailed account on childbirth simulators in past and present.

2.1.1 Fetal head: anatomy and biomechanics

The fetal head plays a crucial role in the biomechanics of childbirth. The interaction of the head with the surrounding maternal structures may is of great importance and has a major effect on the outcome of childbirth. The fetal head shape, which includes its general size and relative proportions which are represented by a number of key obstetric diameters, is thought to be one of the main factors affecting the progression of the baby through the birth canal.

The material properties, which describe the deformation of the head while progressing though the birth canal, are equally important. The deformation is

caused by the pressures of the maternal anatomy on the fetal head and results in the shape variation adequately described by the above mentioned diameters. This phenomenon is commonly known as fetal head moulding. Only a small number of researchers have investigated the particular effects of fetal head moulding on the progression of labour.

Research by McPherson and Kriewall (1980a,b) was aimed at investigating the material properties of the fetal cranial bones with a further intention of developing a model of fetal head moulding during the first stage of labour. The work presented a series of experiments whereby bending tests were used to determine the material properties of the parietal bones of a fetal skull. Further, a low element count FE model of the parietal bones was developed with the determined material properties incorporated. The FE model was validated and shown to exhibit similar behaviour to the tested samples of the real skull bones. Despite its simplicity, the model was a first step towards the description of the deformation of the fetal head during the first stage of labour, when subjected to the pressures of the maternal uterine cervix.

The work by McPherson and Kriewall (1980b) was further extended by Lapeer and Prager (2001). The first improvement was to create an accurate 3D model of the fetal skull. A laser scanner was used to obtain a high-precision 3D model of an exact plastic replica of a real fetal skull. The resulting data was post-processed and a high-quality 3D model (with around 64,000 elements of approx. 1mm in size) of the fetal skull was generated. The model was then used to generate a FE model by incorporating the material properties based on those reported by McPherson and Kriewall (1980b). Intra-uterine pressure and radial cervical pressure models were used to simulate the effect of the cervix on the fetal head during the first stage of labour. All this was combined to obtain a FE simulation of fetal head moulding during the first stage of labour. The results showed good compliance with the clinical results by Sorbe and Dahlgren (1983) who measured the principal obstetric diameters of 319 babies, directly after birth and one day and three days later, to assess the moulding process in an inverse manner.

This research on the biomechanics of the fetal head present a suitable source for the fetal skull models that can be used for childbirth simulation. The fetal head shape used in the childbirth simulation system described in this the-

sis is based on the models produced by Lapeer and Prager (2001) for the fetal head moulding experiments. The model has been laser scanned and then post-processed to generate a high-quality fetal skull model suitable for FEA. Note, that the actual moulding phenomenon is not incorporated into the simulation, but rather the fetal skull mesh geometry is used.

2.1.2 Pelvic floor: anatomy and biomechanics

Another important role in the biomechanics of labour is played by the pelvic floor. The pelvic floor is the main source of the resistive pressures acting on the fetal head. It is reasonable to assume that such a prominent source of interaction with the head can have considerable effect on the movements that the fetal head makes during the passage through the birth canal. Therefore, one of the main tasks of this literature survey was to collect a sufficient number of data on the pelvic floor anatomy and biomechanics. Specifically the data that are particularly useful for the FEM simulation of the pelvic floor as part of our childbirth simulation are of main focus.

A study of pelvic floor biomechanics was presented by Janda et al. (2003). The paper describes the steps taken in order to obtain a geometrical model. The fibre direction of the levator ani muscles are identified with the use of a palpation device. Additionally, they obtained a data set of the pelvic floor structures from MRI images. This information can be combined into a valid FE model that can be used for childbirth simulation. Janda (2006) perform a series of tests that can be used to generate material properties of the female pelvic floor. The paper presents the study of pelvic floor prolapse mechanics and how the material behaviour of the pelvic floor affects it.

d'Aulignac et al. (2005) et al. performed a series of experiments to study the properties of the female pelvic floor. For the pelvic floor geometry they used the point data made public by Janda et al. (2003). The material model used for the simulation was based on the research by Humphrey and Yin (1987) on cardiac muscle tissue tests. The material properties were applied to the geometry constructed from the point data to obtain a FE model. The model was used created to investigate the effects of childbirth on the pelvic floor. The actual

childbirth was not simulated though in the same paper. The use of the cardiac tissue material properties in a childbirth simulation could be a limiting factor in terms of accuracy.

Lee et al. (2005) studied the effects of imposed contractions and strain on the stresses occurring in the pelvic floor. A patient specific model of the pelvic floor was constructed form MRI scans. The variations in the stress values are reported when pelvic floor thickness is varied. The average thickness of the pelvic floor is 3mm. The variations were -10% and +10%. The results indicated that the stresses occurring in the muscles are inversely proportional to the pelvic floor thickness.

Noakes et al. (2008) performed a FE simulation of the pelvic organs during a "maternal bearing down" manoeuvre by applying a pressure on the FE mesh in the downwards direction. The geometries for the several organs present in the simulation was based on the data of MRI scans of live subjects. The material model used in the simulation was a Mooney–Rivlin hyperelastic mode with material constants c10=4.5 kPa and c20=2 kPa. The simulation was performed by gradually increasing the simulated abdominal pressure. The results were validated by comparison to the dynamically captured MRI images. The results were shown to be of adequate accuracy in general when compared visually to the dynamic MRI images, but at the same time errors were present when comparing the deformations of particular tissues, such as the displacement of the organs in the anteroposterior direction.

Rao et al. (2010) undertook the study of pelvic organs prolapse. The bladder, vagina and rectum were simulated using FEA to investigate the pelvic organs prolapse. MRI scans were used to construct the geometries of the pelvic organs. Apart from these organs the simulation also included a model of the maternal pelvic floor based MRI scan data. The elastic modulus of the pelvic muscles (6 MPa) was assumed to be 10 times the elastic modulus of the ligaments (0.6 MPa) as the muscle strength is considered higher by the authors. An simulated increase in the intra abdominal is used to analyse the behaviour of the FE model of the organs to identify the areas undergoing larger strain.

Saleme et al. (2011) presented further developments in pelvic floor FE simulation. They utilized similar methods presented in (Rao et al., 2010). The

geometry was obtained from MRI scans following manual segmentation. The model obtained was used to simulate the deformations of the pelvic floor when voluntary contractions are performed. The main aim of the experiments was the study of the urinary continence mechanism and its maintenance. A FE simulation of the pelvic floor and bladder, urinary tract, vagina and pelvic floor was created. Different degrees of pelvic floor activation were simulated. The contraction degree sufficient for maintaining the urinary continence was the main subject of study.

There has also been research where visco-elastic properties of the pelvic floor were studied (Jing and Valadez, 2011) and (Dejun Jing, James A. Ashton-Miller, 2013). The data was used to perform childbirth simulations and study the deformations imposed on the pelvic floor soft tissues. Data obtained from tests on the pubococcygeus and the iliococcygeus muscles was reported by Jing and Valadez (2011). Jing and Valadez (2011) undertook a series of bi-axial tests performed on a sample pelvic floor sample tissue of squirrel monkey. Later Dejun Jing, James A. Ashton-Miller (2013) created a computer simulation, which takes into account the viscoelastic nature of the pelvic floor muscles, demonstrated that a single volitional push synchronized with the uterine contraction results in more effective progression of the fetus towards birth.

Rubod et al. (2012) conducted mechanical tests on human female cadavers to obtain the mechanical properties of the pelvic floor organs. The tests were conducted on the vaginal tissues, bladder and rectum. The study suggested that the vaginal tissues play a more important role in abdominal organ support than thought before. The findings were then used to suggest better ways of applying prosthetic implants in the vaginal region when performing reconstructive surgeries. The data, however, is only for the tissues listed and no information is provided for the levator ani muscle complex. The importance of the vagina, bladder and rectum is thought to be limited when compared to the effects of the levator ani muscles in the context of fetal head movements (Li et al., 2010), (Hoyte et al., 2008).

Brandão et al. (2015) continued the study of the effects of the voluntary contractions on the state of the pelvic floor. The advancement presented in the paper is the inclusion of the majority of the pelvic floor organs, namely the

bladder, rectum, levator ani, vagina, major ligaments, uterus and pubic bone. The model presented in the paper is notably complete including the reproductive, digestive organs and the connective tissues. The geometries of the organs were obtained from MRI scans of a number of volunteers. Manual segmentation was performed to obtain separated geometries of each of the organs. Most organs are given distinct material properties. The pelvic floor materials are based on data from Janda (2006). The results of the simulation good agreement with the images from dynamic MRI.

Luo et al. (2015) used volunteer MRI scans to obtain the geometry of the pelvic floor. The material properties were taken from the literature. The data published by Janda (2006) was used to fit a hyperelastic material response model. The main aim of the simulation was to study the behaviour of the pelvic floor when intra-abdominal pressure is increased. The results are claimed to be useful for surgical planning. It included anterior and posterior vaginal walls, levator ani muscle, cardinal and uterosacral ligaments, anterior and posterior arcus tendineus fascia pelvis, arcus tendineus levator ani, perineal body, perineal membrane and anal sphincter. The levator ani muslces were simulated with induced impairment to identify how the damaged muscle can affect pelvic organ prolapse and continence.

2.2 Childbirth simulation

The research reviewed above has mainly focused on the the fetal head and pelvic floor anatomy and biomechanics. The next step further in the study of the biomechanics of human labour is the combination of the above studies into a congruent simulation of childbirth. Such a simulation would require the full spectrum of the biomechanical details mentioned above with addition of the interaction aspect between the fetal head and the maternal pelvic floor and bony pelvis. This presents a more challenging task and has been attempted by the research reviewed below.

A number of obstetrics training simulators have been produced over the past decades. They vary in their clinical quality and implementation approach. By quality we understand the fidelity of the simulation with reference to the visual and physical phenomena of a real childbirth. One of the aspects in which the implementation differs is in terms of the tools used for the realization of the

simulator: the traditional mannequin based simulators, software simulations that are used in conjunction with real mechanical apparatus and lastly the purely software-based simulations.

Another classification is proposed by Lepage et al. (2015), where childbirth simulators are subdivided into two classes based on the goal the simulator is used for. The first type are the simulators used only for the purpose of training of obstetricians and midwives. The examples of such simulators are (Moreau et al., 2007, 2008; Buttin et al., 2009). The main aim of such is mostly to serve as demonstrative and training tool. The simulations need not be particulary accurate as long as they demonstrate simulated scene sufficently accurate. The lack of anatomical and physical accuracy, however makes them unusable for scientific studies. These models tend to be simplified and only the general anatomical features are present. The second class is comprised of simulations featuring higly accurate representations of the maternal organs and the fetal body. The anatomy included in the simulation is based on realstic models obtained from scan data. Such simulation systems are better when employed for scientific experimentation, such as studying the effects of childbirth on the pelvic floor or attempting to simulate cardinal movements in a forwards-engineered way. Examples of such systems can be found in (Li et al., 2010; Parente et al., 2009, 2010; Dejun Jing, James A. Ashton-Miller, 2013; Lepage et al., 2015).

Melchert et al. (1995) proposes FEA based approach to the problem. This is an earlier simulation system using FE for the simulation of childbirth. The contact between the fetal body and the maternal organs was considered. The author performed FEA on the whole body of the fetus and took into consideration the maternal soft tissues involved in the process. The simulation accounts for the whole fetal body and the maternal lower-body. The models were created from MRI scan data. The paper indicates having a realistic FE model capable of simulating the full progression of the fetal head through the birth canal. The results presented in the paper are particularly good when considering the early attempt at using FE for childbirth simulation. However, no indication of the cardinal movements is given.

A childbirth simulation using haptic feedback is reported by Kheddar et al. (2004). The system uses imposed trajectories of the fetal descent but the inter-

action of the fetus with the pelvic floor is performed based on a physical model. The behaviour of the soft tissues is simulated using FEA. They use a system coupled with a general purpose haptic device to allow the user to interact with the simulated process. The virtual hand presents an interesting mode to provide a more immersive simulation. The system is limited to simple interactions with the fetal head while calculating the forces for haptic feedback.

The research by Lapeer and Prager (2001) presents a mechanical model for fetal head moulding. This phenomenon occurs during labour and is essentially fetal head deformation due to labour forces. In a further paper, Lapeer et al. (2004) present a system developed to provide forceps delivery simulation with augmented reality (AR). They use tracking markers on the forceps to capture input and simulate the instrumental delivery accordingly. Live video is then combined with rendering of the fetal model to compose an AR simulation. In the same paper they present a mechanical contact model based on FEA to calculate the effect of the contact forces between the forceps and the skull. In both of the discussed research they present realistic models of the fetal head obtained using laser scanning. The head's interaction with the forceps is modelled on a basic contact method principle. The use of AR brings additional attractiveness to the simulation as it has more usability in training applications. The problem with the simulation is that it fails to achieve real-time refresh rates. However, in the latter work they propose potential solutions to the problem, such as implementing reduced FEA techniques and reducing the model complexity.

Lien et al. (2004) simulate the levator-ani stretch and its effect on the pelvic floor muscles. The main finding of the paper is the clear correlation between the stretch of levator ani muscles with the size of the presenting fetal head. The stretch on the pubococcygeus muscles is shown to be the greatest in magnitude. The stretch magnitude is shown to be directly proportional to the fetal head size. The difference with other simulations is that the pelvic floor model is not patient specific but rather manually created based on anatomical literature by recreating the underlying fine structures of the pelvic floor and the vaginal wall. The model of the pelvic floor is claimed to be extensively customizable, whereby certain variations corresponding to age, race, etc can be applied. The effects of these variations are then presented as the result of a vaginal delivery on the pelvic

floor. A simulated trajectory is imposed on the head for it to follow the birth canal axis. The levator ani muscle stretch is recorded along the progression until the head emerges from the birth canal.

Lien et al. (2009) continued their previous research Lien et al. (2004) to incorporate a full visco-elastic model of the pelvic floor. Similar to the previous simulation (Lien et al., 2004) a rigid perfect sphere was used to simulate the fetal head. The main aim of the study is to investigate the effects of varying the way the voluntary maternal contractions can affect the outcome and the duration of labour. The paper investigated the difference between the two approaches the way the mother should perform voluntary pushes. The first "tripple-push" approach requires pushing before, on the peak and after the involuntary uterine contractions. The second approach is defined as pushing at the same time as the uterine contraction reaches its peak. The results showed that when a "tripple-push" approach was used the total duration of the second stage of labour was reduced by 16%, but the total number of voluntary pushes required for delivery was 61% higher than the "peak-push" approach.

Hoyte et al. (2008) developed a simulation also aimed at identifying the effects of childbirth on the levator-ani muscle. Third-party FEA tools were used to perform their analysis. The simulation features an MRI scan based bony pelvis and pelvic floor models. The model representing the fetal head in the simulation is basic, whereby a sphere is used to represent the head and the volume that it sweeps throughout the simulation is used as the fetal body. Similar to Lien et al. (2004) the simulation features a spherical fetal head model and applies an imposed trajectory to simulate fetal descent. The trajectory traced by the sphere is imposed based on observations of real childbirth and reverse-engineering. The role of the pelvis is included for the attachment of the pelvic floor. No other object interacts with it during the simulation. The simulation does not run within real-time rates which makes it unsuitable for obstetrical training applications. The lack of a realistic and customizable model of the fetal head also makes it unreliable for predictive applications.

Martins et al. (2007) created a simulation that does consider cardinal movements. The models for the fetus and the pelvic floor are reused from Janda et al. (2003). The main focus of their research was in identifying the effects of

levator-ani muscle activation during simulated childbirth. The cardinal movements are reverse engineered by imposing rotations on the simulated fetal head. They study the results of cardinal movements on the pelvic floor soft tissues and muscles rather than the opposite., i.e. the reverse effects of the muscles on the fetal head movements and deformations are not considered. The simulations by Martins et al. (2007) and Parente et al. (2010) are not interactive or executed in real-time, which make them inapplicable to training scenarios where real-time interaction between the trainee and the virtual objects is required.

Parente et al. (2008) used FEA to perform analysis on the effects of childbirth on the pelvic floor. The pelvic floor model was created from an MRI scan set by extracting a point cloud representing the required tissues. The point cloud was then used to build a NURBS surface. This resulted in a smooth pelvic floor model, as opposed to direct MRI extracted models which very often contain large amounts of artefacts due to noisy scans. Further work by Parente et al. (2009) reuses the same models and software to study the effects of malpresentation (e.g. occiput-posterior position) of the fetal head on the pelvic floor. The resulting stretch magnitudes are compared between the correct and incorrect presentations. The malpresenting case is shown to have larger stretch values which indicates higher risks of injury. Both studies show interesting simulations of childbirth, but they are focused on the pelvic floor muscles and tissues rather than the cardinal movements of the head. The fetal head movements are not claimed as imposed but little indication of the methods used to obtain them are presented.

A more recent study by Parente et al. (2010) presents further development of the childbirth simulator whereby a FE model for the fetal head is used. The comparative results of the occiput-anterior and the occiput-posterior simulations were revisited. The effects of muslce activation during an occiput-posterior presentation are investigated. The results indicate that introducing muscle activation results in higher stress values, which may indicate more injury in childbirth. Like in their previous work (Parente et al., 2009), the head is forced to follow a given trajectory and rotations are imposed. The simulation featured a whole fetal body model and a model of the fetal skull. The method of force transmission between the external surface and the FE skull is not apparent.

Ashton-Miller and DeLancey (2009) present a FE pelvic floor model in con-

junction with a fetal head model for the purpose of simulating childbirth. The main aim of the simulation is to identify the effects of the childbirth on the pelvic floor. The presented shell based FE model of the pelvic floor is simulated using a visco-hyperelastic material response. The geometry seems to be obtained from manually segmenting a set of MRI scan data, although no clear statement of that is provided. The simulation features a fetal head that is pushed through the pelvic canal and the pelvic floor by an expulsion force reaching 120N. The results of the simulation are used to demonstrate the areas of the pelvic floor that undergo the largest principal stress. No indication is provided concerning the nature of the cardinal movements depicted in the paper. Namely, a part of the extension is visible as the fetal head descends through the pelvic canal but no indication is present describing the mechanics of this movement. Considering the fact that the paper mentions a particular magnitude value for the expulsion force, it is fair to assume that the descent of the fetal head is at least partially unrestricted and physically simulated.

Li et al. (2011) studied the effects of introducing anisotripic behaviour to the pelvic floor muscles. The MRI based pelvic floor geometry is enhanced with the data describing the muscle fiber directions. This model is used to build a FE model with anisotropic behaviour. The aim of the research was to identify how the anisotropicity affects the pelvic floor damage during the second stage of labour. The material properties are based on the ones reported by Janda et al. (2003). The simulation results indicated that the strains in the pelvic floor and the force required to observe delivery is lower when the anisotropic behaviour is incorporated into the FE model.

Dejun Jing, James A. Ashton-Miller (2013) present a patient-specific model of the maternal pelvic floor. They also present a model of the fetal head and maternal pelvis. All models were constructed from MRI scans using segmentation and surface extraction techniques. Constitutive models of visco-hyperelastic materials for FEA were created from data acquired from bi-axial tests on pelvic floor tissues initially reported in their previous research (Jing and Valadez, 2011). External commercial software was used to pre-process the FE models and perform the analysis. The trajectory of the head is imposed based on a reverse-engineered path. A good analysis of the effects of childbirth on the perineal body is pre-

sented. The material properties used in the analysis are based on testing actual pelvic floor tissues.

Yan et al. (2015) have performed a series of childbirth simulations while varying the anatomical measurements of the fetal head. The presented research is closely connected to one presented by Li et al. (2010). The fetal head models have been acquired from 26 newborn skull X-ray CT scans. The maternal pelvis and the pelvic floor was adapted from an MRI scan of a nulliparous woman. An isotropic exponential constitutive relation was used to describe the material behaviour of the pelvic floor. The material properties used were adopted from the work by Janda (2006). The results show compliance with the results of the previous experiments by other authors. The paper mentions that the positions of one of the points on the fetal head has been prescribed so that the main trajectory of the head is imposed making it not fully forwards-engineered. At the same some motion and rotation is allowed for the head so that it can navigate through the birth canal with less resistance. Each of the 26 childbirth simulations took approximately 7 days to complete making the simulation far from real-time. The behaviour of the pelvic floor is modelled with FE soft tissues, whereas the behaviour of the head and maternal pelvis is modelled as rigid bodies. The results from the series of experiments were then used to create a predictive statistical model able to predict the outcome of labour based on the anatomical measurements of the fetal head and maternal features. The created statistical model showed good accuracy when validated, but was limited in terms of further variations of the simulation parameters. While the simulation results were shown to be accurate the simulation rates measured in days.

Lepage et al. (2015) created a numerical model of the majority of the maternal structures that are of importance during the active stage of labour. The main aim of the work presented in the paper is focused on creating accurate numerical models in order to simulate fetal head descent through the birth canal. This makes their work classifiable into the second class of mechanically accurate systems according to their own classification framework. The work is expected to help with understanding the causes of urogenital prolapse by studying the most vulnerable soft tissue elements of the maternal pelvic structures during delivery. MRI scan images of a pregnant patient at 34 weeks gestation were obtained. The scans were

then manually segmented and the raw mesh post-processed to obtain smoother results. Due to the fact that the scans were at 34 weeks gestation the resulting meshes had to be scaled up to match the scales described in the clinical literature. The same approach had to be used for the creation of the ligament and tendon muscles which were mapped from a non-pregnant model to a pregnant one. The material properties of the maternal soft tissues was also taken from non-pregnant subjects but according to Badir et al. (2013) the stiffness of the cervical tissues drops considerably during pregnancy and falls to values 6 times smaller than the values from a non-pregnant cervix. Lepage et al. (2015) applied the same ratio to all other material properties of maternal soft tissues. This generalization could be erroneous, but validation is required to make conclusions. The linear isotropic material model was used to simulate the organ behaviour. The values for the Young's elastic modulus of the pelvic floor is assumed to be 0.06MPa and the Poisson's ratio 0.45. Shell elements were used to discretize the soft tissues as they are all predominantly thin sheet like structures. Lack of other material parameters indicates that the tissues were modelled using isotropic linear elastic material. The pelvis was considered to be fixed and the nodes of the pelvic floor were attached to the corresponding anatomical points on the pelvis. No friction between the fetal head and the maternal soft tissues was considered as the interfaces are thought to be thoroughly lubricated. The motions of the fetal head were modelled by imposing a trajectory based on anatomical literature and the authors' experience. The results reported in the paper show the system being used to simulate childbirth with multiple fetal head sizes. The reported maximum stress locations are indicated as the most vulnerable locations. The locations are in agreement with other works reported in this review (Li et al., 2010), (Lien et al., 2009), (Parente et al., 2009).

The simulation features almost a complete anatomy of the maternal structures making the simulation one of the most advanced at this point in time. The described work presents a considerable advance in computerised childbirth simulation but at the same time it has the disadvantage of an imposed fetal head trajectory. Such imposed trajectories do not necessarily follow the path of least resistance which means that spuriously large stresses may occur in the numerical model of the pelvic floor. This may lead to inaccuracies of the quantitative results

reported but at the same time if only a qualitative observation is made of the location of the highest stress values on the pelvic floor and not the actual stress quantities, then the results are likely to be sufficiently reliable. Additionally, using a linear material model for the pelvic floor may be less accurate, as compared to a hyperelastic behaviour.

Noritomi et al. (2013) presented a childbirth simulation to study the effects of childbirth on the pelvic floor. The simulation featured a basic spherical fetal head that was propelled through the pelvic floor. The pelvic floor was modelled based on manually segmented MRI scans. The regions of high stress were investigated during the simulation. These areas were indicated as the most vulnerable to damage during childbirth. The material properties from Janda (2006) were used for the simulation.

2.2.1 Forward engineered movements of fetal head

Some of the reviewed papers model the fetal head as a perfect sphere (Noritomi et al., 2013; Lien et al., 2004; Hoyte et al., 2008), the rotations of which are irrelevant. When a more realistic model of the head is used (Lepage et al., 2015; Ashton-Miller and DeLancey, 2009; Dejun Jing, James A. Ashton-Miller, 2013), the motions of the head during the second stage of labour are imposed using a predefined trajectory based on observations of real-life labour. There are also simulations that do not use an imposed trajectory. This distinction was described further in Section 1.4.2. One of the earliest research in the area of computer based childbirth simulations is by Geiger (1993). The research is to our knowledge the first one that features an attempt of simulating childbirth whilst also avoiding imposed trajectories in terms of rotations on the fetal head. The simulation was focused on investigating the rotations of the fetal head while it traverses the birth canal. The paper reported that the fetal head successfully navigated the birth canal and was born only in the cases when the rotations resembled the cardinal movements close enough.

The presented simulation investigated techniques for building meshes of models based on the Delaunay triangulation approach. The mesh was built from raw MRI data using manual segmentation by drawing organ contours. After describ-

ing the work concerning generating the meshes, the paper continues to attempt implementing a childbirth simulation. The simulation is based on the compliance principle. The principle is concerned with optimizing the rotation and orientation of the fetal head while it navigates the pelvic canal, by minimizing the contact forces occurring between the head and the pelvis. The contact forces, in turn, are calculated based on a *surrogate* force, which is based on the depth of the penetration of the fetal head and bony pelvis polygonal models. The paper claims that the simulation represents realistic movements of the head. The results present a plausible behaviour for the fetal head. However, the mechanical contact method used is rather simplistic and to a degree lacks a solid physical basis and may therefore not exhibit high accuracy in terms of the fidelity of the cardinal movements. The fact that arbitrary movement and rotation is applied without proper consideration of contact forces renders the simulation unreliable for the purposes of training. Moreover, the simulation may result in an infinite loop of optimizations so that the head will continue oscillating without a trajectory to successfully traverse the pelvis being found. Another important point is that if the infinite loop is not entered the approach will always render the delivery possible. This includes even severe cases of fetal head and maternal pelvis (cephalo-pelvic) disproportion. The effects of the maternal soft tissues on the biomechanics of childbirth are also ignored.

Buttin et al. (2009) present their work where they have managed to model fetal descent during labour without a predefined trajectory. They based the model on physical properties of the main bodies in contact with the fetus and the fetus itself. The simulation involves both the soft and more rigid structures of the maternal pelvic region as the pelvis, pelvic floor, bladder, colon, vagina. The fetus is modelled in a simplistic manner by two rigid ellipsoids for the head and the trunk. The two ellipsoids are enclosed by a soft shell representing all the soft tissues of the fetus. Although the presented simulation is fully forwards-engineered, analyzing the cardinal movements of the head was not an aim. The fetal head and the pelvis were simplified extensively and no fetal head rotations were reported.

The work by Li et al. (2010) is another important contribution to the area of childbirth simulation. The paper presents a childbirth simulation framework

developed to simulate the second-stage of labour based on patient-specific data. The paper describes how MRI scans of the pelvic-floor were used to obtain a FE model suitable for FE simulation. The fetal head model was also generated based on scan data. The main aspect of the paper is related to studying the effects of constraining the motion of the fetal head while it is progressing through the canal of the pelvic floor. Three scenarios were simulated to identify the effects. The first scenario involved fully constraining the bottom (mandibular area) of the fetal head and prescribing a path towards the vaginal opening. The second involved only constraining the position of a single point on the surface of the fetal skull near the occiput and leaving the rotations fully unconstrained. The third scenario has no constraints on the fetal head allowing translations and rotations based on the contact between the skull and the pelvic floor. One of the most important assertions in this paper is made emphasising the importance of studying the cardinal movements of the fetal head without imposed translation or rotation. The paper's findings indicate that the unconstrained simulation generated smaller stresses and strains in the pelvic floor FE model when compared to the ones occurring during a simulation with a fully imposed trajectory. Fully unconstrained head demonstrated smallest effect on the pelvic floor. While there were no prescribed trajectories of the fetal head used, no distinct cardinal movements were observed. In fact, only the translational displacements were reported in the paper. Based on the figures shown in the paper it can be seen that some resemblance of the internal rotation is observed, but closer study is required. The interaction between the fetal head and the pelvic floor was modelled, but no maternal bony pelvis is shown. The lack of a full maternal bony pelvis model could be the cause of the missing cardinal movements. Alternatively, the simulation may well be capable of demonstrating them when a more detailed study is performed focusing on identifying the cardinal movements. When considering the presented simulation framework as a means for real-time training simulations, there are reasons to believe that it may be insufficiently fast. Considering the similarity with the work by Yan et al. (2015), it could be concluded that the simulation times are measured in days, which makes this simulation inapplicable for training purposes.

2.2.2 Mechanical simulators

Purely mechanical simulators

The work by Smyth (1974) demonstrates a hydro-mechanical simulation devices that was used to study the effects of membrane rupture and the change in the intra-uterine pressure on the blood circulation of the fetus. The paper initially mentions the regional practices of labour management in different cultures and even on how the primates manage their own labour. The fetus is represented by a model filled with artificial vessels and is submerged into another vessel filled with a liquid representing the uterus. There is also a liquid moving through the vessels in the fetal body. Various ends of the vessels are connected to pressure measurement devices and readings are taken when the pressure of the external vessel is reduced simulating rapture of the membrane. The main conclusion of the paper is that the artificial surgical rupture of the membrane around the fetus can have adverse effects on the fetal circulation causing fetal tachycardia.

Sellheim (1924) built a model of the fetus for the purposes of simulating internal rotation. The fetus is represented by a mechanically connected rod system with different bending stiffness in different directions. The stiffness values were approximated from the values acquired from performing experiments on live newborns. The tests involved attaching a rubber suction cup on the newborn's head using which the head was flexed and extended while the pull force was measured. It was found that the head resisted the flexion (bending forwards) more than it did the extension (bending backwards). The maternal birth canal was modelled as a bent tube representing the curve of Carus. The author reports the model of the fetus performed an internal rotation when pushed through the model of the maternal birth canal. This results was used to advocate the author's opinion about the causes of the internal rotation. The cause is stated to be the combination of the varying stiffness of the whole fetal body in different directions and the bent shape of the maternal birth canal. The body is said to need to adapt orientation to successfully be born as to be bent in the most flexible direction (extension). According to this view the resulting rotation then would be undertaken by all the parts of the fetus simultaneously, as opposed to the head rotating first and then the trunk.

Hybrid simulation systems

Commercial force feedback devices, for instance The Phantom Omni [1], tend to lack robustness due their maximum allowable forces. These are usually around 10-20N and are too small to simulate forces exerted during childbirth. The delivery forces of a typical case can be as high as 150N (Ashton-Miller and DeLancey, 2009). Due to this limitation several computer simulators use a hybrid approach where the more intrinsic processes of childbirth are graphics based whereas the feedback interface of the system is realized using a dummy model or a mechanical device.

Simulators by Moreau et al. (2008), Buttin et al. (2009) are examples of such hybrid simulation systems. They use a mechanical component, which is represented by the BirthSIM system (Silveira et al., 2004). The BirthSIM system allows having realistic haptic feedback using a mechanical part resembling a parturient woman with a mechanically articulated fetus. The trainees are able to interact with the mechanical model and observe the relevant information on the screen of a connected computer.

Work by Moreau et al. (2007) uses the BirthSIM system, which allows trainee obstetricians to practice forceps delivery. The software component of the system allows them to observe the process from inside representing the positioning of the forceps relative to the fetus inside the birth canal. The simulation system presented in the work provides a suitable solution for forceps delivery simulation purposes. However, the work is more concerned with the labour (uterine) pressures involved in the process. Minimal attention is given to the interaction of the fetal head with the maternal organs.

Work by Dupuis et al. (2009) builds upon the simulation system presented in (Moreau et al., 2008). The simulation system is successfully used in improving the performance of the trainee obstetricians. The main aim of the discussed paper is to track trajectories of the forceps blades applied by the trainees to identify the problematic points and help improve in educating them. The trajectories of the forceps blade applications made by the trainees are compared to the ones performed by expert obstetricians. This presents a valuable tool for trainee per-

[1] http://www.geomagic.com/en/products/phantom-omni/overview

formance evaluation and further training, but the simulation system is (as before) not considering the causes of the cardinal movements and is focused away from the topic of this research using reverse-engineering.

Buttin et al. (2009) present their work where they have managed to model fetal descent during labour without a predefined trajectory. They use FEA for simulating mechanics of the process and couple it with the BirthSIM system. The model presented considers the complex interactions of several important organs, which happen to be in contact with the fetus during labour. Additionally, it presents a realistic uterine contraction force model. It is also capable of simulating the whole process of childbirth continuously and performs FEA based computations of underlying physics. This is a major advantage, which, however, introduces a set of trade-offs. The drawback is the complexity of the computations, which require the system to use simplified representations of the organs and the fetus. Moreover, the bony structures of the bodies are extensively simplified, such that two ellipsoid objects represent the fetal skeleton. The majority of the maternal internal organs were represented by a single entity with little variation in terms of materials assigned. The pelvis is also simplified to the degree where compliance could be lost. In the paper authors do not investigate whether their simulation shows the cardinal movements. The simulation system is capable of generating realistic forces experienced by the fetus as it is born. The complete system of maternal organs though simplified was sufficient to calculate the expulsion forces acting on the fetus. The calculated forces can be used to render the force feedback on the mechanical system connected for training.

2.2.3 Conclusion

The historical works related to childbirth simulation. As it can be expected the majority of these are purely mechanical simulators used mostly for demonstrative purposes. The haptic and physical fidelity of such simulators was of low quality, but could still serve as a good tool to demonstrate the general principles required for a trainee obstetrician or a midwife.

Some of the recent developments in the area of childbirth simulation were also presented. It can be concluded that there are promising developments in the

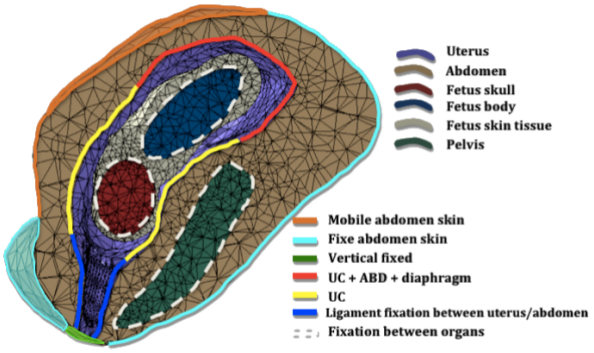


Figure 2.1: Biomechanical simulation of childbirth by Buttin et al. (2009)

field, but more work is required to achieve the high levels of fidelity and comprehensiveness. In particular, none of the reviewed forwards engineered simulations, with the exception of rather unstable simulation of Geiger (1993), managed to show cardinal movements.

The work by Buttin et al. (2009) presents one of the advanced developments in childbirth simulation. However, some of the problems with the work are identified. The most important finding is that such a sophisticated system, but without proper consideration of the bony structures, fails to represent the cardinal movements. It can be argued that this fact supports the proposed idea of the compliance of the bony structures. Therefore, it may be that only a simulation with an accurate representation of the fetal skull and the maternal pelvis will succeed in simulating cardinal movements.

There are many interesting works (Martins et al., 2007; Parente et al., 2008, 2009; Lepage et al., 2015; Yan et al., 2015) that study the effects of childbirth on the maternal pelvic floor. Many of them use highly sophisticated methods such as finite element analysis. Some of these papers can serve as a basis for the pelvic model that our simulation system will eventually use.

It should be emphasized that none of the discussed papers model the cardinal movements in a forwards-engineered way. Some works do take the movements into account to some degree, but mostly as a simple trajectory that is directly applied to the position of the head as opposed to a force pushing the head in the required direction (Lepage et al., 2015). In some cases the trajectory completely

excludes rotations (Dejun Jing, James A. Ashton-Miller, 2013). The work by Yan et al. (2015) presents a hybrid approach where the translation of the head is imposed, but the rotations are allowed depending on the contact conditions between the head and the pelvic floor. Any occurrence of cardinal movements was not reported for such a hybrid simulation. This latter work can be considered as the most advanced in the sense of simulating cardinal movements in a fowards-engineered way.

Most of the works which use imposed trajectories and constraints on the fetal head lack realism in terms of pelvic floor stress values. Admittedly, the analyses performed with constrained head may still present suitable tool for analysing the effects of childbirht on the pelvic floor in the case of some arbitrary fictitious scenario. However, even highly accurate patient-specific skull models should not be expected to yield accurate simulation results. This is due to the fact, as reported by Li et al. (2010), any constraints on the fetal head movements cause spurious increase in the pelvic floor reaction stresses making the whole simulation unreliable. As it will be demonstrated further, the simulation system presented in this thesis aims to fully remove any artificial constraints and imposed trajectories to achieve a more accurate result.

When talking about the classification introduced by Lepage et al. (2015) the work presented in this thesis can be thought of as part of the second class. The main difference from the other representatives of the same class is that the main object of study here is the fetal head and its motions during the second stage of labour. This is opposed to the other works which study the mechanical effects of the fetal head on the soft tissues and structures of the parturient mother. This distinction is mainly made only to emphasise the difference of the proposed simulation system from the ones existing at the time of writing.

The frictional contributions during contact between head and maternal structures are ignored in our simulation system. This choice is based on the fact that the contact surfaces are thoroughly lubricated. There are several previous researches that advocate the same assertion. The research by Rydberg (1935) strongly suggests that the frictional resistance excercies no dominating influence on the movements of the head during labour. His assertion is regarding only the internal rotation though. The works by Li et al. (2010), Yan et al. (2015) and

Lepage et al. (2015) also used frictionless contact for the simulation of childbirth and make a similar assumption.

Another important observation from the surveyed works is that none succeeded in simulating the cardinal movements fully. Many used reverse-engineered kinematics of the cardinal movements, whereas some did not consider them altogether.

A summary of the childbirth simulators reviewed in this literature review is provided in Table 2.1.

2.3 Soft tissue FEA on GPU

2.3.1 Overview

The topic of numerical FEM is a large field of study. There are numerous variations of the method and various approaches to calculating the solution. These variations are often very specialized and are directed at solving a specific type of problems. There are methods suitable for structural mechanics examples of which can be found in (Bathe, 1995). For the purposes of this thesis, we are interested in simulating soft-tissues. Another important constraint on the method and its implementation is that it should be able to be computed at near real-time rates.

Further is an overview of the modern approaches for surgical simulation of soft tissues at near real-time rate.

2.3.2 Realtime FEA

One of the most early works attempting to achieve near real-time soft tissue simulation rates were performed by Terzopoulos et al. (1987). The main aim of the simulation was to achieve high fidelity visuals for graphics related applications.

The ideas used in the previous soft tissue simulations were utilized for FEA in real-time surgical simulation. The works reported by Bro-Nielsen and Cotin (1996) used linear elasticity with the implicit FEM to model deformable tissues. The meshes were made of tetrahedral elements with four nodes. The direct approach of inverting the stiffness matrix and thus calculating the nodal positions is

used. The initial advancement showed good results, but was only suitable for the linear models. No operations that involved changing the topology of the mesh, such as cutting, were possible either.

Bro-Nielsen (1997) report their further work on incorporating 3D solid volumetic FE into surgical simulations. The main aim of the work is to achieve real-time simulation rates for the simulated surgery. A linear elastic model was used to simulate the tissues. The reported results indicate that it was possible to simulate realistic soft tissues for simplistic meshes. An additional requirement to the simulation was being able to simulate cuts by changing the mesh topology. As the authors state the physical accuracy of the simulation was not the primary requirement, but rather the visual fidelity was of more importance. This approximation was mainly allowed to reach higher simulation rates.

Cotin et al. (1999) used a linear elastic model to simulate a human liver. The approach used in the simulation allowed for real-time simulations for tissues that had linear behaviour. The real-time computation rate was possible due to pre-computed elementary deformations that were then used throughout the simulation.

The work by Müller et al. (2002) address the issues that arise when using linear strain measures when performing FEA. The introduction non-linear strain tensor into the analysis fixes the issue, but introduces considerable computational cost. Non-linearity can also lead to numerical instabilities in the solution. The proposed solution result in the simulation having linear costs, but maintaining better fidelity of the simulated results. The principle is to subdivide the strain tensor in the precomputed linear component and a re-computed local tensor of the FE mesh rotations. This field allows us to compute the elastic forces in a non-rotated reference frame while using the precomputed stiffness matrix. It must be noted that the solution accuracy will not be as good as in a non-linear simulation. The reduced artefacts do not guaranty the correct solution, but may help in achieving realistic appearance.

Faraci et al. (2005) proposed using a hierarchical FEM in order to simulate surgical interventions. A high quality mesh is overlaid with a coarser mesh. Adapting the mesh hierarchy in real-time is the main goal of the approach. The real-time simulation is also capable of integrating a force feedback using a haptic

device. The results presented demonstrate the proposed framework using a virtual reality (VR) system.

Irving et al. (2006) attempt to address the issues when large elastoplastic deformations are required to be simulated. The large deformations that may occur in such simulations can lead to degenerate cases, such as collapsed elements the volume of which becomes zero or even inverted. The approach supports any material models and is applicable for both volumetric solids and thin shells such as cloth. The material plasticity is also addressed by the approach, which provides a way to form objects into the desired final shape without sacrificing realistic behavior. The approach can also be extended to arbitrary element types. The details of such extension for the hexahedral elements are described in the paper.

Miller et al. (2007) propose an algorithm is based on the finite element method using the total Lagrangian formulation. This formulation features measuring stresses and strains with respect to the original configuration. The details of this method are discussed in Section 3.2 as it is one of the most important methods for the purposes of this thesis. The method is called the Total Lagrangian Explicit Dynamic (TLED) method. The choice of using the total Lagrangian explicit formulation allows for pre-computing of most quantities that do not change through the solution. The pre-computation is performed once and the computed values are used during the actual solution. This leads to considerable improvement in performance. The use of the Central Difference method for time integration removed the need to use iterative approaches for the integration. The stability of the algorithm is conditional on the time-step used for the time integration. Considering the fact the length of the timestep depends on the simulated material properties, only soft (brain, liver, etc) materials can be simulated in real-time. The simulation accuracy is compared to the modern commercial software and the results indicate that the method is a promising way of accelerating the FE based simulation up to the real-time rates.

Wittek et al. (2005) and Wittek et al. (2007) report their results in performing accurate nonlinear finite element analysis of brain shift. The simulation features several different nonlinear models. A later paper Joldes et al. (2010) is reporting the results of improving the rates of their brain shift simulation. The initial simulation took less than a minute to analyse a 50.000 degrees of freedom model

on CPU. In this paper a GPU based implementation is presented which allows for a 20 times increase in the computation speed. The reported approach allows for real-time surgical simulations for larger meshes. Horton et al. (2010) further investigate implementation of non-linear FEM in surgical simulation.

Taylor et al. (2007) and Taylor et al. (2008) report the development of techniques for high-speed nonlinear finite element analysis for surgical simulation. The proposed approach is based on the TLED method propsed by Miller et al. (2007), as it leads to advantages when simulating soft tissues. The work features a description of the GPU based implementation of the said TLED method. This is potentially the first GPU accelerated implementation of the method. Improved solution speed of up to 16.8 times faster compared with a CPU implementations are reported. The solution accuracy was assessed by comparing the results obtained with a GPU to the ones reported by Miller et al. (2007). Taylor et al. (2009) additionally extended the GPU based TLED implementation to incorporate the visco-elastic anisotropic material behaviour. The presented approach allows for realistic time-dependent anisotropic material behaviour, while still maintaining the considerable performance improvement of parallel GPU execution.

Comas et al. (2008) present a GPU based TLED implementation integrated into the open source SOFA framework (Allard et al., 2007). The implementation described in Taylor et al. (2008) has been adapted to work with the popular SOFA framework. The benefits of this integration are in bringing together the high performance GPU TLED implementation into the benefits of the SOFA framework. The SOFA framework has been shown to be a flexible, extendible and highly modular (Allard et al., 2007) A wide range of surgical simulations can benefit from using a high performance non-linear FEM.

Taylor et al. (2010) described how the reduced order modelling (ROM) in FEA can be used for real-time surgical simulations. The main idea behind ROM is that the full displacement field is reduced to a *generalized displacement* field which has a much smaller dimensionality. The explicit FEM can benefit from ROM when the reduction is applied to the explicit time integration. The traditional issue of the explicit FEM, whereby a small timestep is a required condition for stability, can be alleviated with the introduction of ROM. The required timesteps with ROM explicit FEM become considerably larger than in the full non-reduced

one. Taylor et al. (2011) further investigate the application of ROM's FE in an explicit setting. The results reported in this paper indicated that the performance speed-ups can reach significant levels (more than ten times) while retaining good accuracy with less than 5% reported error values.

Johnsen et al. (2014) present NiftySim, a FE software, that has been designed to allow implementing high-performance soft tissue simulations. The work is based on the advancements in GPU TLED implementation presented by Taylor et al. (2008). The improvements include a generalized framework for soft tissue simulation with support for multiple material models, membrane and shell elements, and contact modelling facilities. The code base mainly consists of C++. The GPU parallelization is achieved using the CUDA framework, which is discussed later in this thesis. Support for contact modelling and non-linear constitutive models is also provided. The paper provides some implementation details of the GPU based TLED implementation and can be of great use.

Strbac et al. (2015) provide a detailed analysis of the GPU based implementation of FEA. The simulation We present an implementation and detailed analysis of an FE algorithm designed for real-time solution, particularly aimed at elasticity problems applied to soft tissue deformation. The TLED approach initially proposed by Miller et al. (2007) is implemented on the GPU. The performance of the GPU implementation is studied and the results are compared to the results obtained using commercial software. It is shown that in conjunction with modern computing architectures, solution times can be significantly reduced, depending on the solution strategy. It is claimed that for some problems it is possible to compute a simple set of test cases 30–120 times faster than compared commercial solutions, while maintaining good accuracy. Strbac et al. (2017) continue studying GPU accelerated TLED approach for simulating human physiology and surgery.

2.4 Computational mechanical contact

Computational mechanical contact is an important topic in surgical simulation. Most surgical simulations will feature objects undergoing contact and this interaction needs to be modelled with as much accuracy as possible. A large body

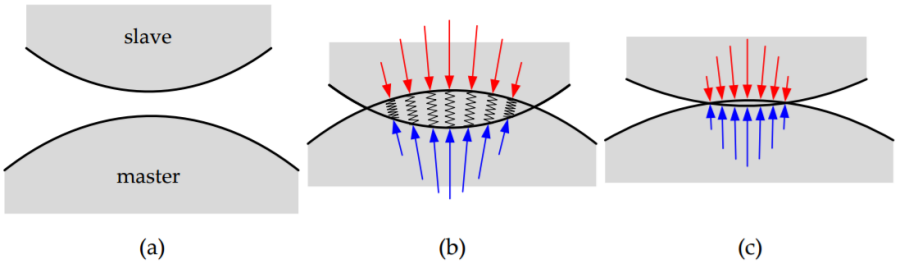


Figure 2.2: Two surfaces in contact. When the penalty method is used, the contact interfaces are connected by a collection of springs pulling the surfaces apart. (a) no contact, (b) penetration occurs and penalty forces (springs) are applied, (c) contact is resolved with small penetration remaining. Figure adapted from Yastrebov (2013)

of research has been dedicated to computation mechanical contact. A broader description of computational contact methods can be found in (Wriggers, 2008).

The next section starts with a broad overview of the existing approaches for computational mechanical contact. Further, the methods used in conjunction with explicit dynamics FE are investigated..

2.4.1 General mechanical contact methods

Penalty based methods

The idea behind the penalty methods is to remove the strict contact constraints of non-penetration by means of a penalization of the constraint violations. Thus is the name "penalty methods". The magnitude of the penalization increases according to how severely the constraint is violated (Kikuchi and Oden, 1988). In fact, it can be interpreted as a series of springs in the contact interface with zero initial length, as represented in Figure 2.2 adapted from (Yastrebov, 2013).

Hallquist et al. (1985) present an application of the penalty method to a 2D contact method. Given two 2D meshes in contact, the contact interface is searched for penetrating nodes. The penetrating nodes are then pushed out of

penetration by applying a penalty force, which is calculated according to

$$f_c = \alpha g \tag{2.1}$$

where f_c – penalty contact force update to the penetrating node, g – gap or penetration that was found between the node and the master surface, α – penalty parameter indicating how strong the penalization force should be.

The penalty approach dictates that the non-penetration condition stated is only approximately fulfilled (Yastrebov, 2013). This is illustrated in Figure 2.2 (c). For the strict fulfilment of the non-penetration condition the penalty parameter α would need to be infinite, which would render the solution impossible. The penalty is also arbitrary and must be determined empirically (Kikuchi and Oden, 1988). The penalty parameter may become inapplicable from one problem to another, when the problem geometry scale, timestep or material properties are changed.

Lagrange Multiplier Methods

An alternative method to a penalty based method is the Lagrange Multiplier Method (LMM). The method essentially fixes the issues arising when using the penalty method. The contact constraints can be satisfied exactly without using an infinite penalty parameter. This method is commonly used in optimization problems to find the maximum/minimum of a functional subjected to equality constraints. The idea behind the LMM is to introduce a vector of additional unknowns λ. These additional variables are called the Lagrange multipliers. The resulting problem is essentially an optimization problem. The solution methods try to iteratively identify the location of the optimal point, which represents the solution. An exhaustive description of the method is provided by Yastrebov (2013) and Wriggers (2008).

The implementation of LMM is based on introducing additional degrees of freedom into the solution. These represent the contact forces between the bodies in contact. Considering that the regular FEA problem is already complex, with the introduction of the additional variables into the system, the solution becomes more difficult to obtain. In fact for relatively large problems pure LMM becomes

impractical to compute (Pietrzak and Curnier, 1999).

Powell (1969) modified the regular LMM approach to address some of its issues. The modified approach is referred to as the Uzawa's algorithm or Augmented Lagrangian Method (ALM).

The main principle of the approach is to combine the advantages of both the penalty and the Lagrange multiplier methods and avoid their drawbacks. The main advantage of the LMM methods was that an exact enforcement of contact constraints is possible without introducing instabilities. This, however, caused major computational difficulties as the exact solution is hard to find. Iterative solution methods can take a long time to find the optimal solution point. By introducing some elements of the penalty method the ALM can be expressed as a Lagrange multiplier formulation regularized by penalty functions Yastrebov (2013). This approach leads to improved solution convergence, thus considerably improving the speed of the solver. Simo and Laursen (1992) further developed the ALM by improving the robustness and introducing frictional constraints.

The ALM approach is thought to be a better alternative for the practical application of computational contact as compared to pure LMM methods (Pietrzak and Curnier, 1999). The issue with computational complexity remains even when using ALM. For simulations requiring real-time update rates and large problems a faster alternative needs to be found.

2.4.2 Contact in explicit dynamic FEA

Force based approaches

Early implementations of contact in an explicit dynamic settings was presented by Taylor (1989). The software featured in the paper is the PRONTO 3D; a three-dimensional transient solid dynamics code. It is capable of analyzing large deformations of highly nonlinear materials subjected to extremely high strain rates. The formulation used is a Lagrangian explicit with explicit time integration. Hexahedral elements were used in the simulation by incorporating a suitable formulation for the element into the FE model. Polar decomposition of the deformation gradient is used to calculate the strains in the unrotated configuration. An accurate incremental algorithm was developed to determine this rotation and

is described in detail. The contact algorithm featured in the software allowed for the impact and interaction of deforming contact surfaces. The contact force calculation was based on the Lagrange Multiplier method.

Heinstein (1997) demonstrated how transient explicit dynamics problems can be modelled in conjunction with contact. The matrix-free approach, where no global stiffness matrices are constructed, allowed for efficient and robust implementation of contact enforcement. The primary contribution being the iterative approach used to enforce the contact non-penetration conditions. In a later paper Heinstein et al. (2000) discusses the treatment of contact—impact modeling in a large deformation explicit dynamic setting. The application of the presented method was mainly in impact modelling but could be adapted for quasi-static contact as well. Frictional problems are also possible to simulate using the approach presented in the paper. The contact detection method presented is based on identifying nodes in contact and their projections on the contact surfaces efficiently in terms of time. Multiple object may be undergoing contact as part of the simulation. The reported computational contact method has served as a basis for further research in mechanical contact in the explicit dynamic setting as discussed further.

The explicit dynamic approach proposed in Miller et al. (2007) can be extended to handle contact between the simulated objects. The initial introduction of contact into the TLED framework was implemented by Joldes et al. (2010) where they incorporated a basic contact model to simulate contact between the brain and the skull. The basic approach for contact detection featured searching for penetrating nodes and translating them out of penetration. This essentially made this a projection based contact method. No contact forces were considered in the model.

The work by Johnsen et al. (2012) built on the ideas proposed by Heinstein et al. (2000) and integrated them into the NiftySim TLED simulation framework (Johnsen et al., 2014). A full contact modelling pipeline suitable for any explicit matrix-free FEM implementations is presented. The proposed approach attempts to address the issues typical for such explicit FE solvers. The issue being that a large number of time steps are required and contact detection and resolution need to be performed as many times. The initial LMM based method of Heinstein

et al. (2000) and Taylor (1989) was extended by adding contact normal smoothing techniques for improved stability. A friction model was also added. The proposed solution is capable of simulating contact between two soft tissues or soft tissues and rigid objects, such as surgical tools.

Johnsen et al. (2015) report further development ontheir contact method. The paper describes the techniques that were used for the contact detection stage. Bounding Volume Hierarchy (BVH) based spatial partitioning approaches were utilized to improve the performance of the contact solution when larger meshes are simulated. A contribution reported by the paper is in the heuristic based BVH reconstruction approach. The normal directions of the faces within the BVH are used to identify if the BVH parts should be refit which leads to improvements in performance. Additionally, edge-edge contact is also introduced to the original method proposed by Heinstein et al. (2000).

Projection based methods

The projection based methods present another alternative when dealing with computational mechanical contact.

The main issue with the penalty method was the exact satisfaction of the contact.

When considering the Lagrangian based methods in the context of an explicit setting such as the one proposed by Heinstein et al. (2000), the solution depends on the length of the time steps used for the explicit time integration. The details of how exactly this is manifested are presented in Section 3.3.5.

The main principle in the projection based approaches is to treat the contact conditions kinematically, that is without considering the contact forces, which in turn will resolve the contact. The contact is resolved by "moving" or "projecting" the violating nodes out of penetration. This approach allows for exact contact resolution with no erroneous penetration remaining and the computational cost of performing the projection is vastly smaller than LMM or ALM approaches. In fact it rivals the simplicity of the penalty based approach.

The issue with the method is that prior or upon projection, the force that should be acting on the interacting objects is unknown. This is not an issue for

the deformable FE object but when the second object is a dynamic rigid body that needs to be pushed away from contact, the "pushing" force is unknown. This is contrary to the penalty method where the force is readily known and used for both the deformable body nodes and the rigid body. The unknown contact force presents difficulty when simulating contact between two deformable bodies. The penetrating nodes and the penetrated surface needs to be "pushed" in amounts appropriate to their masses and stiffness. This requires a well defined force.

The Decomposition Contact Response (DCR) method proposed by Cirak and West (2005) presents an improvement on the basic projection based methods. The enforcement of the contact conditions is performed kinematically, by imposing the displacements on the penetrating nodes. The issue with the projection based methods, where the contact force is not immediately known, is addressed by a momentum based approach. Instead of relying on the contact force in the contact interface, the proposed approach uses momentum exchange between the interacting nodes, subject to interpolation when nodes hit triangles instead of other nodes. These moments are known prior to and during contact as the mass and the velocity of the nodes in contact are known. The method is readily applicable for impact modelling, but may be less relevant for quasi-static problems as there is little momentum to exchange.

Yastrebov (2013) presented a partial Dirichlet-Neumann (pDN) contact method which is also based on the projection based contact resolution. Even though the method presented in (Yastrebov, 2013) is not directly applicable to an explicit dynamics setting, the ideas proposed there are very relevant for the work presented in this thesis and should therefore be covered. The initial step of the contact method is focused on an efficient approach for contact detection to identify the penetrating nodes. The non-penetration conditions are then enforced by projection nodes onto the surface by use of the Dirichlet boundary conditions. The tangential contributions to the contact state, such as friction, are then treated as Neumann boundary conditions. This implies that the friction is treated as a stress and is applied to the penetrating or sliding node as opposed to using a kinematics based resolution. Yastrebov (2013) undertook a detailed analysis of his proposed method and found that it is applicable for wide range problems. The advantages of the presented work are that it is capable of efficiently resolving the

contact exactly without inheriting the computational complexity of LMM and ALM, as those increase the number of unknowns that need to be found.

The treatment of the contact conditions as a set of boundary conditions is an important concept for the projection based contact presented as part of this thesis in Section 3.3. It does not seem that the scenario where the rigid body in contact with the deformable is dynamic, in the sense that the contact should be able to affects its movement. For the purposes of the childbirth simulation in this thesis the pDN method is extended to deal with soft vs dynamic rigid object contacts in an explicit dynamics setting. The issue whereby the contact with dynamic rigid body was not possible due to the unknown contact force is addressed by our approach and the solution is presented in the next chapter.

Table 2.1: Summary of existing childbirth simulations. LE - linear elastic material, HE NH - Neo-Hookean hyper-elastic material model, HE MR - Mooney-Rivlin material model, VHE - Visco-hyperlasitc, ANI - anisotropic material models.

	pelvic floor material	fetal model	deformable head	realtime	imposed trajectory	cardinal mov.
Melchert et al. (1995)	LE	full	no	no	no	no
Geiger (1993)	no	head	no	no	no	no
Lapeer et al. (2004)	no	full	yes	yes	no	no
Kheddar et al. (2004)	LE	full rigid	no	yes	no	no
Lien et al. (2004)	HE NH	sphere	no	no	yes	no
Martins et al. (2007)	HE	full FE	yes	no	yes	no
Hoyte et al. (2008)	HE NH	sphere	no	no	yes	no
Parente et al. (2008)	HE	full FE	yes	no	yes	no
Parente et al. (2009)	HE	full FE	yes	no	yes	no
Buttin et al. (2009)	HE NH	full simple	yes	yes	no	no
Dupuis et al. (2009)	HE NH	head simple	yes	yes	no	no
Ashton-Miller and DeLancey (2009)	VHE	head	no	no	yes	no
Li et al. (2010)	HE MR	head	no	no	hybrid	no
Li et al. (2011)	ANI HE MR	head	no	no	hybrid	no
Dejun Jing, James A. Ashton-Miller (2013)	VHE	head	no	no	yes	no
Gerikhanov et al. (2013)	no	head	no	yes	hybrid	partial
Yan et al. (2015)	ANI HE MR	head	no	no	hybrid	no
Lepage et al. (2015)	LE	head	no	no	yes	no

Chapter 3

Methodology

3.1 Mesh Generation

3.1.1 Overview

This chapter is dedicated to the issues related to FE mesh generation. The general principles of generating and pre-processing FE meshes is discussed along with the specific treatment of the custom-built mesh geometries used in the childbirth simulation.

The next sections describe the specific meshes featured in the childbirth simulation and their sources of the raw data and the methods for their generation.

An important aspect of mesh generation should be considered when working in the context of computer simulation. The models used in the simulation should be of great accuracy, but at the same time have minimal geometrical complexity in terms of number of primitives. The requirement for lower complexity is due to the fact that very complex models are difficult in processing by numerical methods implemented on computers. The greater processing power of modern computers allows for higher mesh complexities, but in the presence of multiple complex organ meshes their interaction becomes expensive to model. The meshes should be simplified while keeping the approximation error within small margins. In depth coverage on the to the topic of mesh enhancement can be found in (Hansen et al., 2005).

It is also important to keep the mesh quality at a higher level. Apart from

suffering computational complexity, the numerical methods suffer from numerical errors and degenerate cases. Collapsed tetrahedra and poor aspect ratios can lead to poor accuracy and stability of most numerical methods. The poor quality mesh primitives need to be pre-processed to obtain improved results.

The methods for detecting, visualizing and treating poor quality primitives are presented in the following sections. The results of pre-processing are then demonstrated after which they are passed to the volumetric mesh generation routines.

3.1.2 The maternal bony pelvis

The bony pelvis model used in the simulation described in this thesis was created from the Visible Female [1] CT scans The raw scanned data had to be extensively processed and simplified to arrive at a model usable in a computer simulation.

The techniques employed to rectify the model were remeshing and decimation. The pelvic mesh generated from the Visible Female CT data contained a large number of defects in the form of poor quality triangles. A large amount of spurious data was also introduced when generating the mesh from the scan data. The data had to be manually processed to remove the spurious parts of the mesh. Additionally, octree based re-meshing had to be applied to improve the quality of the valid parts of the geometry which exhibited poor quality.

Decimation was primarily required due to the overly complex result of the remeshing. The original octree-remeshed meshes had a large number of triangles, i.e. around the 60k mark. Most of the triangles in the mesh are spurious and do not represent the actual geometry. After decimation was applied the number of triangles was reduced to 14k.

The current childbirth simulator features a rigid pelvis without potential deformation. The model is also only used for attachment of the deformable soft tissues and for contact with the fetal head. This suggests that not all of the parts of the pelvis will be in contact with the fetal head at any point in time. Most of the pelvis is also not attached to any soft tissues. These parts of the pelvis

[1] The Visible Human Project is an initiative from the Library of Medicine in Bethesda, Maryland, US. http://www.nlm.nih.gov/research/visible/

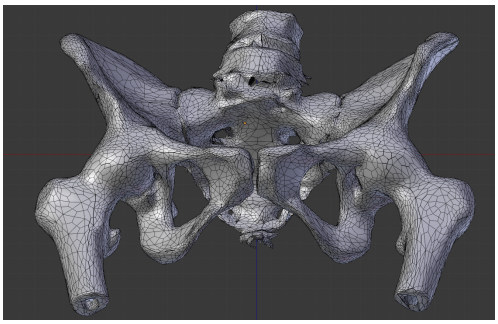


Figure 3.1: The original model of the maternal pelvis obtained from Visible Human scan data.

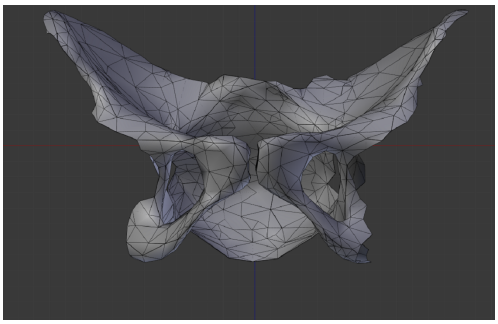


Figure 3.2: The mesh of the pelvis after cropping. The mesh was pre-processed to reduce its complexity and improve quality. The areas of the mesh that never come into contact with any part of the fetus are removed.

that have no physical relevance to the simulation can be ignored and removed from the mesh. This simplification is beneficial to the simulation complexity and speed of computation.

3.1.3 Fetal skull model

A suitable model of the fetal skull was produced by Lapeer and Prager (2001). The mesh was used to simulate the phenomenon of fetal head moulding. The model is shell mesh consisting of approximately 64k triangles. The total number of triangles was reduced to 7.9K triangles by applying decimation.

The cavities under the zygomatic bones were removed and the surface smoothed. Considerable smoothing was applied to the skull to emulate the effect of soft tissues and fascia surrounding the bony skull. The resulting smoothness of the mesh is also beneficial for the contact method implemented for the purposes of the simulation.

The dimensions of the head were modified to match the values of a typical fetal head at full gestation.

The original skull model (Lapeer and Prager, 2001) also missed the mandible. It is justified to remove the bottom part of the skull including the mandible for the

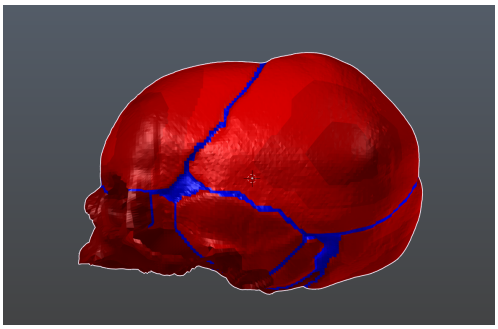


Figure 3.3: The original fetal skull mesh obtained from Lapeer and Prager (2001).

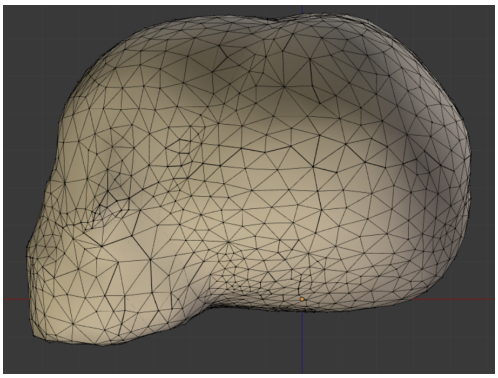


Figure 3.4: The mesh of the fetal head after pre-processing. The mesh was pre-processed to reduce its complexity and was smoothed. The mandible is added to the original mesh.

purposes of simulating fetal head moulding. The only areas noticeably affected by the compressive forces of the cervix and other soft tissues are solely located on the top of the skull around the occipital area and the bregma.

The purely physics based simulation such as the one attempted relies fully on the contact between the various models. The resulting interaction is strongly dependent on the geometries of the models involved. It can be seen that the missing bottom part of the skull and the mandible in particular can have severe effects on the fidelity of the simulation. The actual experimental results are reported in Section 5.5.

Figures 3.3 and 3.4 shows the original mesh and the pre-processed mesh with the smoothing applied to the surface of the mesh and the added mandible.

3.1.4 Pelvic floor structures

Research reviewed in Section 2 often used MRI scans from various origins to generate pelvic floor geometries. Several research groups used the Visible Female [1] MRI scans to generate the pelvic floor meshes. However, it should be taken into account that these scans were taken from a middle-aged woman. The reproductive

[1] https://www.nlm.nih.gov/research/visible/visible_human.html

system at such an age undergoes considerable transformation (Zaki et al., 2013) and could be unrepresentative of a typically younger parturient woman. Further indication of the effects of the age on the mechanisms of labour are given by Arulkumaran (2016). The same study by Zaki et al. (2013) indicates that even within the time between 20 years and 40 years there is a decrease in the duration of the first and second stages of labour further indicating that the physiology undergoes a change with the age of the mother.

Additional difficulty is presented by the fact that manual segmentation of the pelvic floor from MRI scans is highly complex (Lepage et al., 2015). Even in the case of their simulation the scan data were of a 34 week pregnant woman, which required them to upscale the resulting segmented meshes to full term (40 weeks) size. This emphasises the complexity of obtaining an accurate model of the maternal and fetal structures at full term. Additionally, the complexity of obtaining the pelvic floor model through segmentation of MRI scans forced the researchers to resort to manual recreation of the pelvic floor structures using anatomical descriptions. This same approach had to be taken for the ligament and tendon models.

The work by Mitsuhashi et al. (2009) is an overview of the BodyParts3D anatomical mesh library system. The on-line service described in the paper presents a convenient way to search an anatomical database created as part of the TARO project (Nagaoka et al., 2003). The search system allows finding any of the available pre-segmented organ models based on various criteria. The original data scans were taken from the TARO projectz, which was aimed at constructing a whole-body MRI scan collection of average Japanese male and female bodies. The MRI scan data is represented by 2mm slices from several volunteer subjects. The scans were then manually segmented as part of the BodyParts3D project. The missing information was manually clarified using anatomical text books and professional clinicians were used as consultants for complicated segmentation cases. One of the major differences of this project when compared to the other anatomical databases is the fact that most of the meshes are provided in a pre-segmented and pre-meshed format. This alleviates the need for manual segmentation on behalf of users but at the same time may present difficulties if the quality of the pre-segmented meshes is insufficient. The access to the original

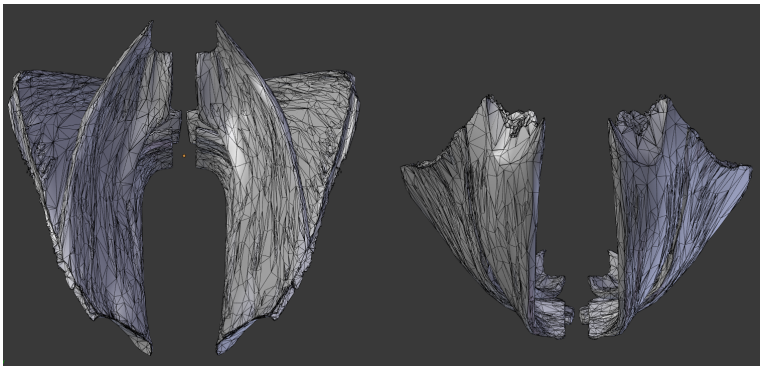


Figure 3.5: The transverse view of the obtained mesh of the pelvic floor muscle complex (left). The sagittal view of the same meshes.

volumetric scan data is not provided which complicates the manual segmentation and corrections for the end user.

There is also another effort focused on constructing a whole-body computable scan databases of the male and female bodies. The one presented by Shin et al. (2013) is aimed to create a whole-body model of a Korean individual. The Visible Korean project was also used in work by Shin et al. (2013), where the main aim is the creation of an accurate model of the female pelvic region. The scan data was manually segmented and then used to generate a 3D meshes for use in computer simulations.

The original pelvic floor meshes

The most accessible and permissive of the above listed sources is the one reported in (Mitsuhashi et al., 2009). Additionally, most of the meshes provided in the database are already pre-segmented. Moreover, not all of the above listed sources were readily available, while the chosen option provided an easy to use online interface for downloading the required meshes. Additionally, the meshes of the pelvic floor substructures are readily available as separated sub-meshes.

The combination of the unmodified sub-meshes of the pelvic floor obtained from Mitsuhashi et al. (2009) is shown in Fig. 3.5

Octree based remeshing

The initial mesh obtained from the source at Mitsuhashi et al. (2009) is of poor quality when considered for FE mesh generation. The topic of mesh quality of FE meshes is a large one and a considerable amount of literature is dedicated to it. For the purposes of this thesis we will only cover the specialized techniques applicable for the particular case of the pelvic floor FE mesh.

There is wide range of methods that allow remeshing (Marchandise et al., 2010). The most commonly used approach for meshing when a new mesh needs to be generated from some non-mesh data is the marching cubes algorithm (Lorensen et al., 1987) and its variations. The source data is typically represented by an analytical function or a volumetric data set, but a pre-existing mesh can be used as a source as well. Remeshing is concerned with creating a mesh from an existing mesh that has defects where the new one is expected to improve upon. The marching cubes algorithm is also applicable for such remeshing cases, where the existing mesh needs to be modified to obtain a higher quality mesh.

There are many metrics of calculating the quality of triangles in a mesh (Pebay and Baker, 1991). The triangle quality is calculated as the ratio of the radius of circumscribing circle over the longest side according to Eq. 3.1.

$$Q = \frac{R}{L_m} \tag{3.1}$$

where Q – triangle quality measure, R – radius of circumscribing circle, L_m – longest edge of the triangle.

Many applications require the triangles to be of a specific shape in order to prevent them from running into numerical problems, e.g., numerical simulations based on FEM. For this purpose, triangles are required to have the ratio of the radius of the circumcircle to the shortest edge to be close to 1. The quality of triangles can be inspected by rendering a mesh in wireframe mode or hidden-line image mode. An alternative and efficient solution is to colour code the quality of the mesh based on triangle aspect ratios. With such a colour coded mesh it is trivial to identify a single triangle with poor aspect ratio. Figure 3.7 shows a close-up view of a part of the whole pelvic floor mesh focusing on the puborectalis muscle. It can be seen on Figure 3.6that the triangular mesh quality is generally

poor. seen.

Combining meshes

As we have already seen in Section 1.2, the pelvic floor mesh is subdivided into separate substructures. They in turn are closely connected to each other and ultimately form a unified object which can be treated as a single structure. Based on this premise the next step of mesh generation was to combine all the meshes of the pelvic floor into a single structure.

The separate meshes were available as listed below:

1. left puborectalis (LPR)
2. left pubococcygeus (LPC)
3. left illioccygeus (LIC)
4. right puborectalis (RPR)
5. right pubococcygeus (RPC)
6. right illioccygeus (RIC)

The meshes that represent each of the substructures had overlapping areas which could be used as a point of connection when combining them. To perform the actual combination, a Boolean union operation was applied. After the meshes were combined, a connection was manually introduced between the two sides of the generated unified pelvic floor muscle emulating the perineum (Figure 3.9). The mesh of LPC and LPR are fused with the corresponding parts of RPC and RPR meshes. The perineum is positioned according to obstetric and anatomical references (Drake et al., 2014). The connection also implicitly results in a realistically positioned anal opening of the pelvic floor.

The results of the above manipulations are demonstrated in Figure 3.8

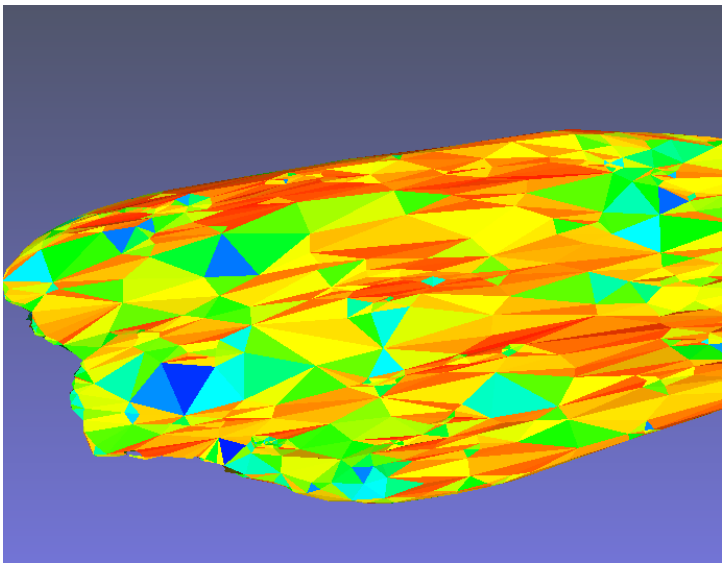


Figure 3.6: The original mesh obtained from Mitsuhashi et al. (2009). Large red regions on the surface indicate a large number of triangles with poor aspect ratio.

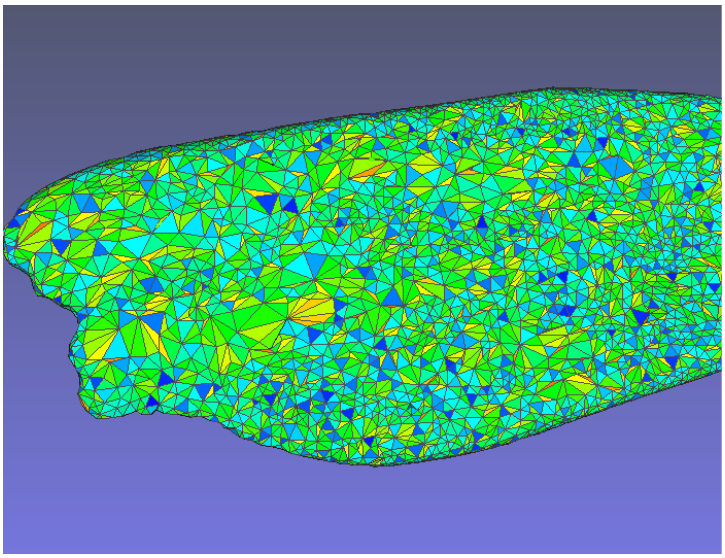


Figure 3.7: The pelvic floor mesh after octree based remeshing. The mesh contains a larger number of triangles, but their aspect ratios are predominantly good as indicated by largely blue surface colour.

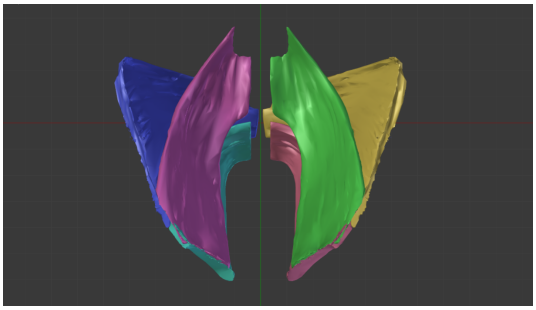


Figure 3.8: The original disjoint pelvic floor substructures obtained from Mitsuhashi et al. (2009). Each differently coloured mesh is a distinct mesh representing: green - LPC, yellow - LIC, red - LPR, pink - RPC, blue - RIC, cyan - RPR.

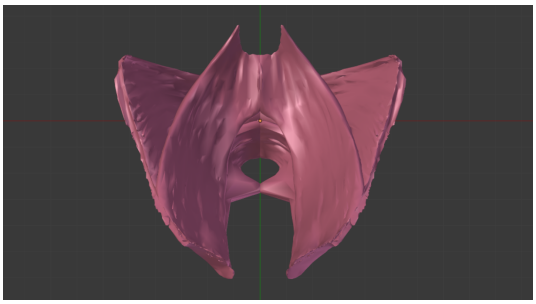


Figure 3.9: The pelvic floor mesh after combination using a boolean union operation. The perineal body is also introduced whereby the LPC, LPR, RPC and RPR are fused at a point.

Laplacian smoothing

The main principle of Laplacian smoothing is averaging. Each of the vertices of a given mesh are positioned at the average position of all the neighbouring vertices. The neighbouring vertices are defined as all the vertices connected to the vertex by the incident edges. The average is calculated using the following equation:

$$\bar{x}_i = \frac{1}{N} \sum_{j=1}^{N} \bar{x}_j \tag{3.2}$$

where $\bar{x}$ – the new smoothed position of vertex i, N – number of neighbours, $\bar{x}_j$ – j-th neighbour position.

The process can be visualized as seen in Figure 3.10. The outermost edges of the neighbouring triangles can be thought of as a single polygon. The vertex is placed in the middle of the polygon, thus producing a smoother mesh (Lapeer, 1999). The process is repeated for all the vertices in the mesh.

Laplacian smoothing is often used for mesh quality improvement in a wide variety of applications (Field, 1988) due its simplicity, relatively low computational costs and good smoothing results. It is, however, not capable of remedying all types of mesh quality issues.

The smoothness of the meshes is important for the stability of the contact

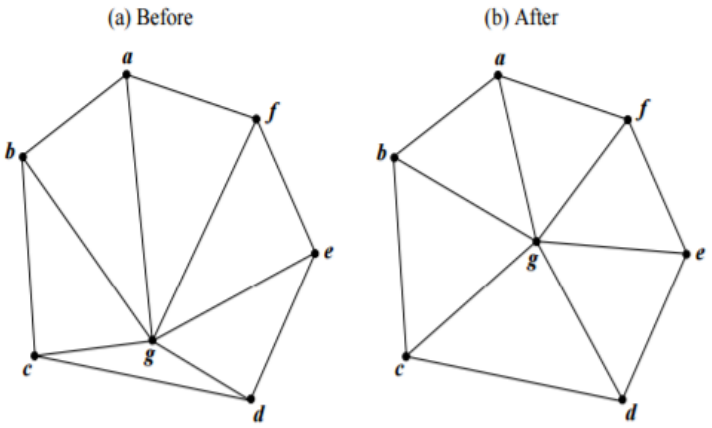


Figure 3.10: Application of Laplacian smoothing on a vertex within a polygon represented by the outermost edges. The vertex is placed in the middle of the surrounding polygon which results in a smoother mesh when the process is repeated for all mesh vertices (Lapeer, 1999).

methods used to simulate interaction between them as discussed further in Section 3.3. In order to improve mesh smoothness and normal orientation continuity we applied Laplacian smoothing.

The issue with traditional mesh smoothing techniques is that their application tends to remove any fine details of the mesh described by high curvature. Such details can include fine ridges, spikes, and protrusions. In the case of holes they typically tend to increase in size. The assumption we adopted for the purposes of our simulation system is that the fine details of the mesh are unimportant for the fidelity of the simulation. Therefore the smoothing applied to the pelvic floor mesh is acceptable

Warping

The meshes obtained as the basis for the pelvic floor mesh used in the simulation of labour are of an adult male. It can be seen that the main anatomical structures of the pelvic floor muscles are common for male and female. The main difference lies in the perineal triangle and the area of the vaginal opening.

It must be noted that the material properties of the pelvic floor muscles are

not the same for a male and female either. This distinction is still respected in our case as we only use the meshes for geometry whilst the material properties are obtained from the literature on female pelvic floor biomechanics.

Keeping in mind these considerations, the model of the pelvic floor of the male needs to be transformed into a shape appropriate for a female. This can be achieved by using various existing warping techniques which allow to transform meshes to desired shapes according to a specified set of constraints and mathematical functions.

For the purposes of warping the pelvic floor mesh we chose to use a basic linear warping technique that uses the distance to the origin of the transformation as the fall-off parameter. This has the effect that the vertices further away from the warp origin are moved less than the ones nearer the warp origin.

The male pelvic floor needs to be warped to match a female equivalent. This can be achieved within a margin of error by warping it to match a female pelvis. The fitting of the pelvic floor mesh to the pelvic mesh can be achieved by ensuring that the pelvic floor attachment points are at the correct locations on the bony pelvis mesh. The corresponding attachment point pairs from the pelvis and the pelvic floor were identified according to the anatomical literature (Drake et al., 2014). Each of the attachment points on the pelvic floor becomes a warp region origin $\vec{O}$. The lines connecting each pair is the warp origin translation $\vec{T}$. The magnitude of the translation is inversely proportional to the distance of the transformation origin. The warped position $\vec{P}_n$ is calculated according to:

$$\vec{P}_n = \vec{P}_{n-1} + \vec{T}\frac{|\vec{P}_{n-1} - \vec{O}|}{R} \tag{3.3}$$

where $\vec{P}_n$ - position of the vertex after warping, $\vec{P}_{n-1}$ - original vertex position, $\vec{T}$ - the warp origin translation, R - warp effect maximum radius

The warping was performed using the Blender 3D software (Blender Online Community, 2015). The results of the applied warping are demonstrated in Fig. 3.14. It can be seen that the pelvic floor and pelvis attachment points are the ones that have been transformed the most. The rest of the transformation follows a linear fall-off function showing little to no change in the mesh the further the vertices are. The fall-off radius for the warping was 8cm, with vertices further

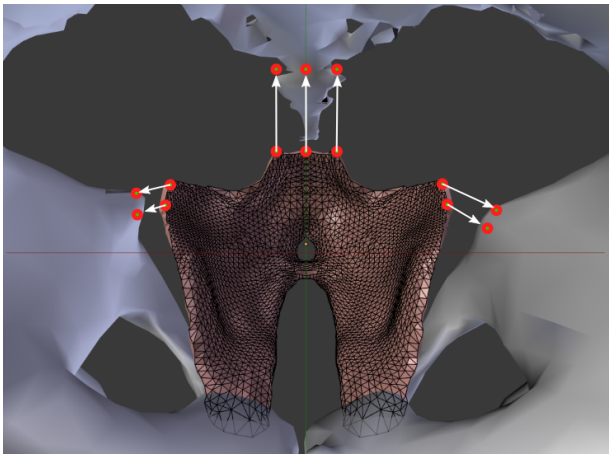


Figure 3.11: The pelvic floor mesh demonstrating the points of interest. Each of these points is shifted towards a corresponding attachment point on the pelvis effectively warping it to fit it.

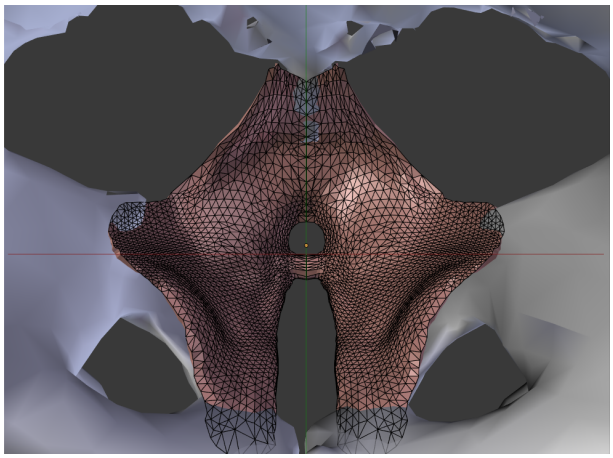


Figure 3.12: The pelvic floor mesh after warping. The muscle attachment points are shifted to the pelvis.

than the radius being unaffected by the movement at all.

Uniform thickness

The original pelvic floor mesh consisted of a number of parts of which each has a different thickness. The mesh parts overlap in some areas contributing to the total thickness of the mesh at the overlapping regions.

Most of the FEM simulations featuring a pelvic floor model reviewed used a uniform thickness model (Hoyte et al., 2008), (Parente et al., 2008), (Parente et al., 2009), (Lien et al., 2004), (Lien et al., 2009), (Li et al., 2010). In fact the uniform thickness is dictated by the fact that shell elements are used to represent the mesh of the pelvic floor.

The benefits of using a variable thickness mesh of the pelvic floor becomes questionable when considering the origins of the raw pelvic floor mesh used in our simulation and the pre-processing applied to it. The regions of higher or lower thickness may have been erroneously misplaced as the result of applying the warping procedures. The male pelvic floor may have considerable difference in the substructures and their muscle thickness when compared to a female one.

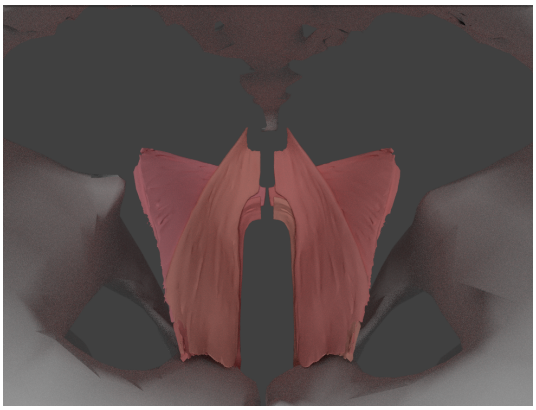


Figure 3.13: The original model obtained from Mitsuhashi et al. (2009).

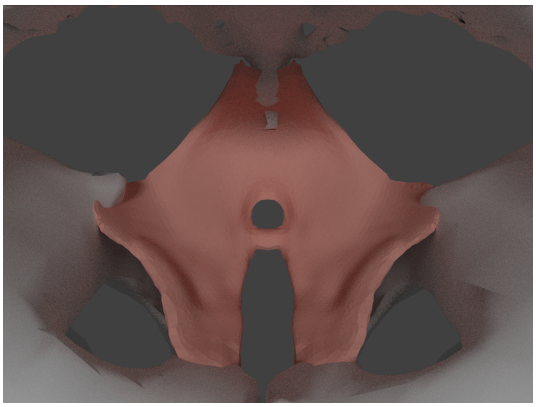


Figure 3.14: The pelvic floor model after remeshing, adding of the perineal body and linear warping to match the dimensions of the female pelvis.

Admittedly, the additional information about the varying thickness could be beneficial for the accuracy of the simulation but considering the potential errors it was deemed unnecessary. Simplifying the mesh to have a uniform thickness is considered to bring more benefits compared to the introduced simplifying approximation.

In order to be generate a uniform thickness mesh the surface of the mesh needs to be extracted. For this purpose all the triangles of the pelvic floor having normals orientated upwards and inwards are selected. The selected region then represents the internally facing surface of the pelvic floor that will come into contact with the fetal head. The rest of the triangles are removed. All the selected triangles are extruded in the anti-normal direction. The inverted direction is chosen in order to avoid artificially increasing the resistance from the pelvic floor. The extrusion thickness is chosen to be 5mm, which is the average between the different levator ani regions reported by Downing et al. (2007).

Volumetric mesh generation

All of the previous steps described in this section were performed on the surface meshes. The surface meshes only describe the out-most shell of the object they

represent. The TLED FE method used for the childbirth simulation requires a volumetric description of the meshes. This implies that the meshes are not hollow, but are made out of a chosen type of volumetric element. For the purpose of our simulation, the tetrahedral finite element description is chosen. This implies that the surface meshes need to be converted to a volumetric representation made of tetrahedral volumetric elements.

The concept of Delaunay triangulation allows generating a high quality triangular mesh out of a 2D contour or a set of points. The idea of a regularized generation of a mesh from a simpler shape can also be extended to the domain volumetric tetrahedra. The "hollow" surface mesh becomes the empty contour. The contour is then "filled" with volumetric elements in a regularized fashion similar to the 2D Delaunay triangulation.

The main principles of volumetric tetrahedral mesh generation from a surface mesh are presented by Shewchuk and Shewchuk (2002). The described approach is based on the concept of the Delaunay triangulation which is extended to the domain of 3D tetrahedral elements.

Labelle and Shewchuk (2007) present an alternative approach to Delaunay based tetrahedral mesh generation. The described mesh generation method is capable of producing good quality tetrahedral meshes with good dihedral angles which is the main measure of a tetrahedron's quality. The process of filling the surface mesh with the tetrahedra is named "stuffing" in the paper.

CGAL The Computational Geometrical Algorithms Library (CGAL) is a large collection of various computational geometry algorithms in C++ [1]. The library is split into packages each dedicated to a specific computational geometry area.

The part of CGAL dedicated to 3D tetrahedral mesh generation is presented in (Jamin et al., 2016). This module is based on Delaunay triangulation and tetrahedralization methods.

The library presents a rich API for configurable mesh generation. The implementation is clear and highly optimized. Rich documentation has a large number of scientific references to the work on which the implemented features are based. However, the generalized nature of the library introduces a set of drawbacks. The

[1] http://www.cgal.org/

generalized nature of the library makes it applicable to a wide variety of problems, but at the same time introduces considerable complexity. It is also relatively hard to incorporate into existing software due to its size and third party dependencies.

Gmsh with NetGen NetGen is a standalone application specifically designed for mesh generation and pre-processing [1]. The theoretical underpinnings of the methods implemented in the NetGen software are provided by Schöberl (1997). The software was tested as part of the Gmsh (Geuzaine and Remacle, 2009) [2].

Gmsh implements its own mesh optimization procedure to enhance tetrahedral mesh quality by means of basic mesh primitive manipulation. The NetGen tetrahedral mesh optimization can be applied on top of the native optimization of Gmsh. The resulting mesh has improved quality.

The Gmsh software has considerable advantages for FE mesh pre-processing. It is even capable of being integrated with various known FE solvers. In conjunction with NetGen it presents a highly effective tool for FE mesh generation and pre-processing. The disadvantage is that it is difficult to integrate into existing C++ simulation software.

TetGen The TetGen Delaunay (Si and Hang, 2015) tetrahedral mesh generation software is publicly available [3]. The quality of the generated tetrahedral mesh is good and can be parametrically controlled. The software is capable of functioning as a stand-alone application with its own console based interface. Additionally, the software is easily integrated into an application as a third party library.

When considering the quality of the generated mesh and the API's ease of use, TetGen appeared to be the preferred solution. It was used to generate all the volumetric meshes used for the FE simulation of childbirth.

The general API for using TetGen is implemented as a function that takes an input structure and generates an output structure. The input structure is populated with the data from the surface mesh, which are the list of surface vertices and the list of triangles represented by vertex indices. The output structure

[1] https://sourceforge.net/projects/netgen-mesher/
[2] http://gmsh.info/
[3] http://tetgen.berlios.de

contains a list of triangles represented by a series of four vertex indices. It also contains a list of vertices that comprise the generated volumetric mesh.

Note that the list of vertices from the output is different from the vertex list populating the input structure. This is the result of additional vertices being added to the original surface list. This leads to the input triangle list being incompatible with the output vertex list. When the triangles are visualized using the original vertex indices and the new output vertices the resulting mesh will look completely different from the original surface mesh.

For the purposes described in Sec. 3.3.2 there is a need for the mesh generation method to provide the mapping between the triangles of the "shell" surface and the generated volumetric mesh. This feature is provided by TetGen as part of the API. The output structure provides an identical list of triangles to the original surface mesh, with the exception that the vertex indices of these triangles are pointing to the new output vertex list. This feature allows for easy mapping between the triangles of the surface mesh with the mesh of the generated volumetric mesh. Additional detail on this feature and motivation is provided in the dedicated section 3.3.2.

3.2 Total Lagrangian Explicit Dynamics on GPU

3.2.1 Overview

This section is dedicated to describing the FEM and specifically its TLED variation, which was used in our implementation. The theoretical introduction to FEM is given with the highlights of the advantages and disadvantages of various formulations. The choice of the integration approach for the solution of the resulting PDE's is then discussed. Later, the formulation of choice is discussed in more detail. Finally, the technical details of our GPU based implementation of the approach are provided.

The Finite Element Method (FEM) is a numerical method extremely useful in a wide variety of physical simulations. It finds applications in mechanical structural analysis, electrodynamics, hydraulics and many others. Such a wide range of applications arises from the fact that FEM can be used in approximately

solving any *field* problem (Hutton, 2004). A field problem, also known as a *boundary value* problem, is a mathematical problem in which a set of dependent variables must satisfy a differential equation everywhere in the specified domain and satisfy the boundary conditions. *Boundary conditions* in a general sense allow enforcing specific values. There are different orders of boundary conditions based on the order of the field variable derivative they are prescribed for. In the context of computational structural FEM, they can be interpreted as a predefined set of values that the displacement field should be equal to at specific points of the analyzed domain.

The main focus of this research is on mechanical structural analysis and particular in soft tissue simulation. Oden (2010) identifies two key attributes of FEM:

Partitioning

The analyzed domain is partitioned into a set of small non-overlapping domains over which a complex function can approximated by a local polynomial function.

Weak-formulation

The local functions are not required to satisfy the analytical solution locally, as long as the overall domain integral satisfies the boundary conditions.

The finite sub-domains are also known as *elements*. The subdivision principle is illustrated in Figure 3.15. For a detailed introduction to the FEM the reader is referred to Hutton (2004).

The governing equation that the method is aimed to solve is:

$$\boldsymbol{M\ddot{u} + C\dot{u} + K(u)u = R} \tag{3.4}$$

where $\boldsymbol{M}$ - mass matrix, $\boldsymbol{\ddot{u}}$ - second derivative of the deformation field, $\boldsymbol{C}$ - damping matrix, $\boldsymbol{\dot{u}}$ - first derivative of the deformation field, $\boldsymbol{K(u)}$ - stiffness matrix, u - nodal deformation field, $\boldsymbol{R}$ - external forces.

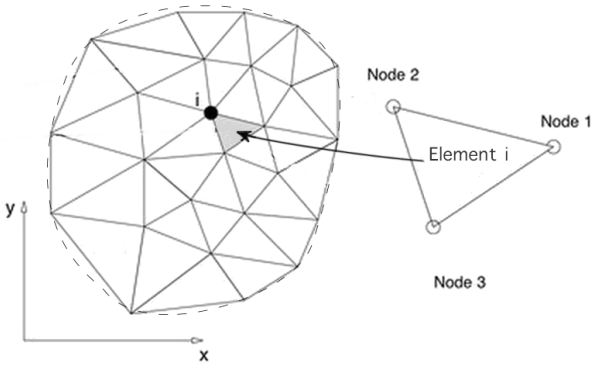


Figure 3.15: Finite element mesh partitioning example in 2D.

3.2.2 Hyperelasticity

For a hyperelastic material model the stress-strain relation is defined in terms of the Hermholtz free-energy function W (Holzapfel, 2000). In the case when the function is only dependent on the deformation gradient $\boldsymbol{F}$ the function is called strain-energy function $W = W(\boldsymbol{F})$. Strain-energy function can also be expressed in terms of various other strain measures. The right Cauchy-Green strain tensor $\boldsymbol{C}$ can be used for evaluating the strain-energy function value $W(\boldsymbol{C})$.

The relationship between the first Piola-Kirchhoff stress $\boldsymbol{P}$ and the deformation gradient $\boldsymbol{F}$ is defined by the partial differentiation of the strain-energy function with respect to the deformation gradient:

$$\boldsymbol{P} = \frac{\partial W(\boldsymbol{F})}{\partial \boldsymbol{F}} \tag{3.5}$$

where $\boldsymbol{P}$ - first Piola-Kirchhoff stress, $W(\boldsymbol{F})$ - strain-energy function, $\boldsymbol{F}$ - deformation gradient.

For a non-linear analysis with finite deformations it is convenient to use the second Piola-Kirchhoff (SPK) stress S and Green strain $\boldsymbol{E}$ (Belytschko et al., 2014). The infinitesimal change of strain energy of different strain measures must give equal values:

$$\partial W(\boldsymbol{E}) = \partial W(\boldsymbol{C}) = \partial W(\boldsymbol{F}) \tag{3.6}$$

The relationship between SPK and the Green strain is similar to the conjugate relationship in 3.5 and is given as:

$$\boldsymbol{S} = \frac{\partial W(\boldsymbol{E})}{\partial \boldsymbol{E}} \tag{3.7}$$

where $\boldsymbol{S}$ - second Piola-Kirchhoff stress

Note the relationship between the Green-Lagrange $\boldsymbol{E}$ and right Cauchy-Green $\boldsymbol{C}$ strains:(Holzapfel, 2000)

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{I}) \tag{3.8}$$

where $\boldsymbol{I}$ - a 3 x 3 diagonal unit matrix.

The relationship from Eq. 3.8 can be used along with the chain-rule to arrive at the second Piola-Kirchhoff stress evaluated in terms of strain-energy function dependent on the right right Cauchy-Green strain:

$$\boldsymbol{S} = 2\frac{\partial W(\boldsymbol{C})}{\partial \boldsymbol{C}} \tag{3.9}$$

3.2.3 Eulerian and Lagrangian formulations

There are two notable approaches to FE discretization based on the choice of coordinate system which is used for the formulation. The Eulerian and Lagrangian formulations use the *spatial* and *material* coordinates respectively. The *spatial* coordinates describe the location of a point in space. The *material* coordinates labels a particle within the modelled object.

The Eulerian formulation is particularly useful in fluid flow modelling. In structural mechanics, typically Lagrangian formulations are used.

Figures 3.16 and 3.17 adopted from Belytschko et al. (2014) demonstrates the visual interpretation of the difference between Eulerian and Lagrangian formulations. The nodes of mesh with Lagrangian formulation remain coincident with the material points after deformation. Contrary to this, the Eulerian formulation will keep the node positions fixed in space, whereas the material points will move.

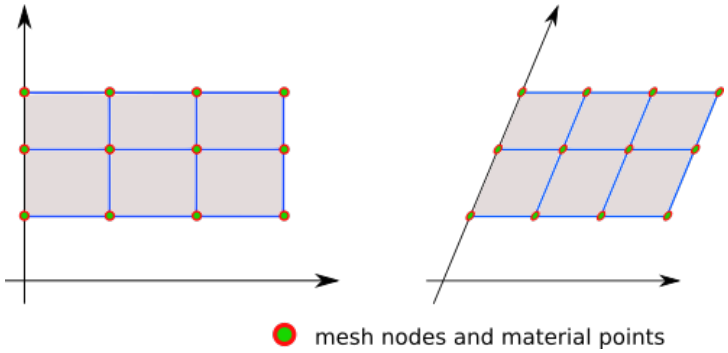


Figure 3.16: A 2D block undergoing shearing described using the Lagrangian formulation. The nodes remain coincident with the material coordinates.

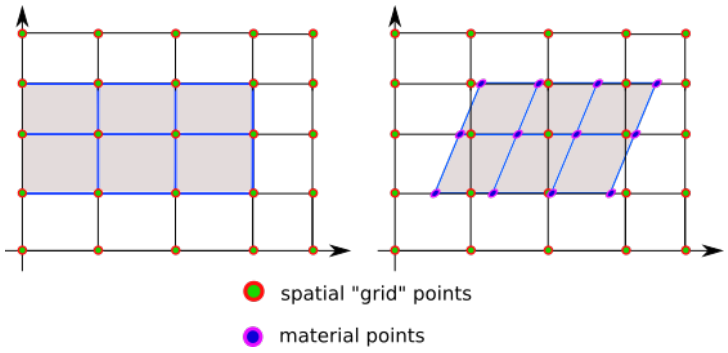


Figure 3.17: A 2D block undergoing shearing described using the Eulerian formulation. The mesh nodes are unchanged, whereas the material points move.

Updated and Total Lagrangian formulations Both the updated and total formulations are Lagrangian in the sense that they express the dependent variables as functions of the material coordinates.

The differences between the two formulations are described by Belytschko et al. (2014). In the updated Lagrangian formulation the derivatives of the dependent variables are evaluated with respect to the spatial coordinates and the current deformed configuration is used to describe the variables. In the case of a total Lagrangian formulation the initial configuration is used to describe the variables. The derivatives of the dependent variables are taken with respect to the material coordinates.

There are also differences in the stress and deformation measures which are typically used in these two formulations. For example, the total Lagrangian formulation customarily uses a total measure of strain, whereas the updated Lagrangian formulation often uses a rate measure of strain.

Although the total and updated Lagrangian formulations are superficially quite different, it can be shown that the underlying mechanics of the two formulations is identical; furthermore, expressions in the total Lagrangian formulation can be transformed to updated Lagrangian expressions and vice versa.

Note that the main advantage of the Total Lagrangian formulation is that a considerable number of values can be pre-computed and then reused throughout the simulation, thus saving computation time.

Total Lagrangian Explicit Dynamic formulation

TLED formulation is a variation on the Lagrangian formulations of the FEM. It was initially proposed by Miller et al. (2007) as a means for efficient computer based finite element analysis (FEA). The alternative approach is known as the Updated Lagrangian Explicit Dynamic approach which is based on using the previously calculated configuration of the deformed body in order to calculate the stresses occurring in the finite elements. In contrast to this technique TLED uses the original reference configuration to perform the calculations. TLED has the advantage of allowing pre-computations of a large chunk of data as compared to the approach using the updated Lagrangian.

The TLED formulation is chosen as the basis of our FE implementation. This approach is described by Miller et al. (2007) and the strong and the weak formulations are presented by Belytschko et al. (2014).

In the TLED formulation the derivatives of free variables are taken with respect to the material coordinates and the required integrations are performed over the reference configuration, which is the undeformed initial configuration of the continuous body.

The FE approximation of motion with help of shape functions is given by:

$$x_i(\boldsymbol{X}, t) = x_{ij}(t) N_j(\boldsymbol{X}) \tag{3.10}$$

where x_i - is the i-th component of the position in the current configuration, x_{ij} - i-th component of the j-th node, $N_j(\boldsymbol{X})$ - shape function for node j. The equation implies summation over j.

One of the most important governing equations for the TLED formulation is the law of conservation of linear momentum (Belytschko et al., 2014) expressed in Eq. 3.11. Note the subscript zeros indicating the initial reference configuration.

$$\nabla_0 . \boldsymbol{P} + \rho_0 \boldsymbol{b} - \rho_0 \ddot{\boldsymbol{u}} = \frac{\partial \boldsymbol{P}}{\partial X} + \rho_0 \boldsymbol{b} - \rho_0 \ddot{\boldsymbol{u}} = 0 \tag{3.11}$$

where ∇_0 - nabla operator with respect to material coordinates, here used in the context of the gradient of the second order tensor $\boldsymbol{P}$ which is the first Piola-Kirchhoff stress, ρ_0 - density in the reference configuration, $\boldsymbol{b}$ - body forces, $\ddot{\boldsymbol{u}}$ - acceleration as the second derivative of the displacement $\boldsymbol{u}$.

The Principle of Virtual Work (PVW) can be described as the *weak form* of the law of the conservation of linear momentum from Eq. 3.11. It is obtained by multiplying the equation by a test function $\delta u(X)$ and taking the integral over the whole domain Ω_0. The integral must evaluate to zero for the momentum to be conserved:

$$\int_{\Omega_0} \delta u (\nabla_0 . \boldsymbol{P} + \rho_0 \boldsymbol{b} - \rho_0 \ddot{\boldsymbol{u}}) d\Omega_0 = 0 \tag{3.12}$$

The integral can be split into components based on their physical interpreta-

tion:

$$\delta W^{int} - \delta W^{ext} + \delta W^{kin} = \int_{\Omega_0} \delta u \frac{\partial \boldsymbol{P}}{\partial X} d\Omega_0 + \int_{\Omega_0} \delta u \rho_0 \boldsymbol{b} d\Omega_0 - \int_{\Omega_0} \delta u \rho_0 \ddot{\boldsymbol{u}} d\Omega_0 = 0 \tag{3.13}$$

The first term from from Eq. 3.13 denoted as δW^{int} is the virtual internal work of the internal forces. The virtual internal work can also be expressed as the product of the internal forces with the corresponding virtual displacements. We can also move δu outside the integral which leads to the following equalities:

$$\delta W^{int} = \delta u f^{int} = \delta u \int_{\Omega_0} \frac{\partial \boldsymbol{P}}{\partial X} d\Omega_0 \tag{3.14}$$

Using the arbitrariness of δu from Eq. 3.14 we can find the the formula for the internal force f^{int} by:

$$f^{int} = \int_{\Omega_0} \frac{\partial \boldsymbol{P}}{\partial X} d\Omega_0 \tag{3.15}$$

The derivative of stress can be approximated with the introduction of the shape functions from Eq. 3.10 thus we obtain the discrete equation for the internal forces:

$$f = \int_{\Omega_0} \frac{\partial N}{\partial X} \boldsymbol{P}^T d\Omega_o \tag{3.16}$$

where f - nodal force, N - shape functions, X - nodal positions in reference configuration, Ω_0 - integration domain in reference configuration.

By combining the shape function derivatives into a matrix denoted as $\boldsymbol{\partial h}$:

$$\boldsymbol{\partial h} = \frac{\partial N}{\partial X} = \begin{bmatrix} \frac{\partial N_1}{X} & \dots & \frac{\partial N_n}{X} \\ \frac{\partial N_1}{Y} & \dots & \frac{\partial N_n}{Y} \\ \frac{\partial N_1}{Z} & \dots & \frac{\partial N_n}{Z} \end{bmatrix} \tag{3.17}$$

where N_i - element shape functions, n - total number of nodes in the element, X, Y and Z - material coordinates.

In matrix form Eq. 3.16becomes:

$$f = \int_{\Omega_0} \boldsymbol{\partial h} \boldsymbol{P}^T d\Omega_o \tag{3.18}$$

The relationship between the first and second Piola-Kirchhoff stresses can be utilized in order to express the nodal force expression in Eq. 3.18 in terms of the second Piola-Kirchhoff stress $\boldsymbol{S}$:

$$f = \int_{\Omega_0} \boldsymbol{\partial h} \boldsymbol{S} \boldsymbol{F}^T d\Omega_o \tag{3.19}$$

where $\boldsymbol{\partial h}$ - matrix of shape function derivatives, $\boldsymbol{F}$ - deformation gradient matrix, Ω_0 - integration domain in reference configuration.

The evaluation of integrals over element domains is not performed.

Note that the strain-deformation matrix $\boldsymbol{\partial h}$ is defined with respect to material coordinates and remains constant with time. This time-invariant matrix can be assumed constant and pre-computed thus saving computation time. The time-dependent deformation gradient $\boldsymbol{F}$, however, needs to be recalculated at each step.

The TLED scheme provides very accurate and efficient simulations for soft-tissue simulation. However, as Miller et al. (2007) described it, it is more suitable for simulating "very soft" tissues. In cases when the stiffness of the material is higher the minimal time-step decreases quadratically. This increases the number of sub-steps that need to be performed to achieve a converging solution. Extra sub-steps lead to dramatically decreased simulation rates.

This characteristic of the TLED method makes it infeasible to use as an approach for simulating bony structures present in childbirth. The fetal head and maternal pelvis deformations cannot be performed using the TLED approach within the required time constraints. Additionally, it is possible that certain tissues in the pelvic floor may have large Young's moduli making it impractical to simulate them using TLED.

3.2.4 Linear tetrahedral element for TLED

Nodal force integration with Gaussian Quadrature

The nodal force in an element is calculated using Eq. 3.19. For the purpose of computer simulation, it is not feasible to evaluate the integral over the element domain analytically. Therefore, it is required to use a numerical approximation. The process of numerically evaluating an integral is called *numerical integration* (NI). One of the most used approaches to NI is Gaussian quadrature (Weisstein, 2016) which in one dimension has the general form of:

$$\int_{-1}^{1} f(\xi)d\xi = \sum_{Q=1}^{n_Q} w_Q f(\xi_Q) \tag{3.20}$$

where $f(\xi)$ - continuous function of natural or element coordinates ξ, n_Q - number of quadrature points used, w_Q - quadrature weight, ξ_Q - quadrature point coordinates.

The quadrature equation for the three-dimensional case is

$$\int_{\Omega} f(\xi)d\Omega = \iiint_{-1}^{1} f(\xi)d\xi d\eta d\zeta = \sum_{Q_1=1}^{n_{Q_1}} \sum_{Q_2=1}^{n_{Q_2}} \sum_{Q_3=1}^{n_{Q_3}} w_{Q_1} w_{Q_2} w_{Q_3} f(\xi_{Q_1}, \eta_{Q_2} \zeta_{Q_3}) \tag{3.21}$$

where $f(\xi)$ - continuous function of natural (or element) coordinates ξ, n_Q - number of quadrature points used, $w_{Q_1}, w_{Q_2}, w_{Q_3}$ - quadrature weights in each dimension, $\xi_{Q_1}, \eta_{Q_2}, \zeta_{Q_3}$ - quadrature point coordinates in each dimension.

The triple summation is often omitted for a single summation where the quadrature weights are multiplied to result in a single weight (Belytschko et al., 2014):

$$\overline{w}_Q = \prod_i w_{Q_i} \tag{3.22}$$

$$\int_{\Omega} f(\xi)d\Omega = \sum_{Q=1}^{n_Q} \overline{w}_Q f(\xi_Q) \tag{3.23}$$

In case of the nodal force calculation the integration is performed on the stress

values over the element domain as seen in Eq. 3.19. However, to apply Gaussian quadrature it is convenient to transform the integral from element domain to the idealized canonical form expressed with respect to material coordinates.

The differential over the domain $d\Omega_o$ can be expressed in terms of differentials $d\xi$, $d\eta$ and $d\zeta$ according to Nanson's formula (Holzapfel, 2000). The Jacobian determinant J is equal to six times the volume of the undeformed tetrahedron, V_0.

$$J = \det(\boldsymbol{J}) = \det(\frac{\partial X}{\partial \xi}) = 6V_0 \tag{3.24}$$

$$d\Omega_o = \frac{1}{6} J d\xi d\eta d\zeta = V_0 d\xi d\eta d\zeta \tag{3.25}$$

$$f = \iiint_0^1 V_0 \boldsymbol{\partial h S F}^T d\xi d\eta d\zeta \tag{3.26}$$

The Gaussian quadrature approximation is then evaluated using Eq. 3.19 and using single summation form from Eq. 3.22 resulting in the following equation

$$f = \sum_{Q=1}^{n_Q} \overline{w}_Q{}^0 V \boldsymbol{\partial h S F}^T \tag{3.27}$$

According to the Gaussian quadrature weight table (Belytschko et al., 2014), for the case of a single point quadrature the single weight w is equal to 2 in one dimension. However, note that our integration is over the range [0, 1], as opposed to the canonical [-1, 1], which scales the weights to be equal to 1. The combined weight $\overline{w}_Q$ is then a product of weights for all three dimensions and is also equal to 1. This results in Eq. 3.27 for the single point quadrature becoming:

$$f = V_0 \boldsymbol{\partial h S F}^T \tag{3.28}$$

where f – matrix of nodal force contributions, V_0 – initial element volume, $\boldsymbol{\partial h}$ – strain deformation matrix, S – SPK, $\boldsymbol{F}$ – deformation gradient.

Note that the shape function derivatives are with respect to the material coordinates (Lagrangian) and the undeformed reference configuration (total) volume is used in the calculation, which indicates that Eq. 3.28 is the TLED formulation

of the nodal force.

Shape function derivatives for a linear tetrahedron

The deformation gradient $\boldsymbol{F}$ matrix can be expressed as:

$$\boldsymbol{F} = \boldsymbol{\partial h u} + \boldsymbol{I} \tag{3.29}$$

where $\boldsymbol{\partial h}$ – shape function derivatives matrix, $\boldsymbol{u}$ – nodal displacement matrix, $\boldsymbol{I}$ – identity matrix

The shape function derivatives with respect to material coordinates are given by:

$$\boldsymbol{\partial h} = \frac{\partial N}{\partial X}. \tag{3.30}$$

It is trivial to evaluate shape function derivatives with respect to the natural coordinates $\frac{\partial N}{\partial \xi}$, as opposed to evaluating in terms of material coordinates. Therefore, by using the chain rule we get:

$$\boldsymbol{\partial h} = \frac{\partial N}{\partial \xi}\frac{\partial \xi}{\partial X} = \frac{\partial N}{\partial \xi}\left(\frac{\partial X}{\partial \xi}\right)^{-1}. \tag{3.31}$$

For a tetrahedral constant-strain element the shape functions are defined as follows:

$$\begin{aligned} N_1(\xi,\eta,\zeta) &= \xi, \\ N_2(\xi,\eta,\zeta) &= \eta, \\ N_3(\xi,\eta,\zeta) &= \zeta, \\ N_4(\xi,\eta,\zeta) &= 1-\xi-\eta-\zeta. \end{aligned} \tag{3.32}$$

where N_i - shape functions, ξ, η, ζ - natural coordinates within an element.

Taking derivatives with respect to natural coordinates and collecting into a

matrix gives:

$$\frac{\partial N}{\partial \xi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -1 & -1 \end{bmatrix} \tag{3.33}$$

The Jacobian matrix relating undeformed nodal coordinates to natural coordinates $\boldsymbol{J}$ can be calculated from the matrix of nodal coordinates $\boldsymbol{X}$:

$$\boldsymbol{J} = \frac{\partial X}{\partial \xi} = \boldsymbol{X}\frac{\partial N}{\partial \xi} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -1 & -1 \end{bmatrix} \tag{3.34}$$

After multiplication the resulting matrix contains the edges of the tetrahedron converging at the fourth node:

$$\frac{\partial X}{\partial \xi} = \begin{bmatrix} x_1 - x_4 & x_2 - x_4 & x_3 - x_4 \\ y_1 - y_4 & y_2 - y_4 & y_3 - y_4 \\ z_1 - z_4 & z_2 - z_4 & z_3 - z_4 \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix}^T \tag{3.35}$$

where E_i - are column vectors representing tetrahedron edges.

$$\boldsymbol{\partial h} = \frac{\partial N}{\partial \xi}\left(\frac{\partial X}{\partial \xi}\right)^{-1} = \frac{\partial N}{\partial \xi}\begin{bmatrix} E_1 & E_2 & E_3 \end{bmatrix}^{-1} \tag{3.36}$$

The determinant of $\det(\boldsymbol{J})$ is equal to six times the tetrahedron volume in the reference configuration. Note that the inverse of $\boldsymbol{J}^{-1}$ contains information about each of the opposite faces from each node.

$$\frac{\partial \xi}{\partial \boldsymbol{X}} = \boldsymbol{J}^{-1} = \frac{1}{6V}\begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \tag{3.37}$$

3.2.5 Time Integration

Equation of motion for the total Lagrangian formulation describes the equation according to which the simulated system behaves. In order to simulate the dynamic problem the PDE from Eq. 3.38, which describes the equation of motion, needs to be solved. The solution involves time integration to find the state of the physical system at a given time t. The solution can be achieved through two different approaches, namely the implicit and explicit time integration techniques.

$$\boldsymbol{M}\ddot{\boldsymbol{u}} + \boldsymbol{C}\dot{\boldsymbol{u}} + \boldsymbol{K}(\boldsymbol{u})\boldsymbol{u} = \boldsymbol{R} \tag{3.38}$$

where $\ddot{\boldsymbol{u}}$ - second derivative of the deformation field, $\boldsymbol{C}$ - damping matrix, $\dot{\boldsymbol{u}}$ - first derivative of the deformation field, $\boldsymbol{K}(\boldsymbol{u})$ - stiffness matrix, u - nodal deformation field, $\boldsymbol{R}$ - external forces.

Implicit solvers

Implicit integration schemes for linear transient problems are unconditionally stable. At the same time, for the wide variety of problems addressed by the implicit integration in practice the stability is conditional. Large time steps can decrease the robustness of the Newton and Newmark methods. However, generally the time steps required for implicit integrators can be significantly larger than those required by the explicit methods Belytschko et al. (2014).

The solution times are also dependent on mesh quality and complexity (number of DOF's). The type of material simulated is also a factor affecting the time required to find the solution but the actual material parameters are irrelevant. This feature is one of the main advantages of the implicit solvers.

The disadvantage of implicit solvers comes from the trade-offs of the qualities described above. The implicit formulation requires constructing a large matrix representing the stiffness of the whole assembly. The size of the matrix is determined by the number of DOF's in the assembly, $N_{dof} \times N_{dof}$. When larger problems are required to solve, the sparse storage techniques are incorporated into the solver (Bathe, 1995). A static analysis may take several minutes, hours or days to compute depending on the size of the mesh, the complexity of the constitutive equations and the boundary conditions.

Another issue with the implicit approach appears when dealing with non-linear FE problems. The case of a linear problem can be calculated directly by stiffness matrix inversion and multiplication with the external loads to achieve the nodal displacements. In the case of a non-linear geometry problem, an iterative numerical method is required to find the solution.

The Newton-Raphson method is a fast and popular numerical method for solving nonlinear equations, as compared to the other methods, such as direct iteration. In principle, the method works by applying two steps (Krenk, 2009)

1. perform an equilibrium check to find if the solution is reached within the desired accuracy

2. if equilibrium is not reached make an adjustment to the deformations as described in Eq. 3.39

An initial guess for displacements is needed to start the iterations. The updated displacements are calculated according to

$$x_{t+1} = x_t - \frac{f(x_t)}{f'(x_t)} \tag{3.39}$$

where x_{t+1} – next estimated solution, x_t – current estimated solution, $f(x_t)$ – the integrated displacement function, $f'(x_t)$ – first derivative of the function.

Explicit solvers

Explicit schemes rely on iteratively evolving the system. The solver updates the system from its initial configuration to arrive at the correct solution.

There are several explicit integration schemes but we will only look at the Central Difference Method (CDM). This method is also known as the explicit Verlet time integration method. Where the explicit Euler integration can be seen as the "forward" difference integration, the Verlet is "central" difference integration approach.

The equation of motion for a dynamic structural problem at time n, solved

by direct integration, is defined as:

$$\frac{1}{\Delta t^2}\boldsymbol{M}u_{t+1} = F_t - R_t + (\frac{2}{\Delta t^2}\boldsymbol{M} - \frac{1}{\Delta t}\boldsymbol{C})u_t + (\frac{1}{\Delta t^2}\boldsymbol{M} - \frac{1}{\Delta t}\boldsymbol{C})u_{t-1} \tag{3.40}$$

Damping needs to be introduced into the simulated system in order to maintain numerical stability of the explicit methods. The Rayleigh type damping is used when dealing with explicit FE simulations (Johnsen et al., 2014). After introducing Rayleigh mass proportional damping and applying CDM, the equation for calculating the nodal displacements for the next simulation step is defined as

$$u_{t+1} = \frac{\boldsymbol{R_t} - \boldsymbol{F_t} + \frac{2}{\Delta t^2}M_i u_t + (\frac{1}{2\Delta t}\boldsymbol{C} - \frac{1}{\Delta t^2}\boldsymbol{M})u_{t-1}}{\frac{1}{\Delta t^2}\boldsymbol{M} + \frac{1}{2\Delta t}\boldsymbol{C}} \tag{3.41}$$

where u_{t+1} – next nodal displacement, u_t – current mass matrix, R_t – external forces, F_t – internal forces, M_j – nodal mass, C – damping matrix.

The above equation can be simplified when considering that most of the calculations are on constant quantities which can be pre-computed. This pre-computation approach is also used by (Johnsen et al., 2014) in NiftySim. Additionally, when the damping value is constant for each node the damping matrix $\boldsymbol{C}$ can be represented by a scalar damping value c. The quantities that are precomputed:

$$\begin{aligned} A &= \frac{1}{\frac{1}{\Delta t^2}\boldsymbol{M} + \frac{1}{2\Delta t}c} \\ B &= \frac{2}{\Delta t^2}\boldsymbol{M}A \\ C &= \frac{1}{2\Delta t}Ac - \frac{B}{2} \end{aligned} \tag{3.42}$$

With the introduction of the pre-computed quantities 3.42 the CDM Equation 3.41 becomes:

$$u_{t+1} = A(\boldsymbol{R_t} - \boldsymbol{F_t}) + Bu_t + Cu_{t-1} \tag{3.43}$$

The convergence on the correct solution is conditional in the sense that the time steps used are required to be sufficiently small. The time step length for a

given simulation is both dependent on the smallest dimension of the elements and the material parameters. Using the central difference time integration scheme, the critical time step is limited by the speed of sound in the simulated material system, c,(Hughes, 2012). The time step must remain below a certain critical value Δt_{crit}, otherwise the mesh will diverge. The critical time step for the central difference method can be estimated from the Courant-Friedrichs-Lewy (CFL) (Lewy H., 1928) condition. The equation for calculating the critical timestep is:

$$\Delta t_{crit} = \frac{L}{c} \tag{3.44}$$

where L – smallest element extent, c – speed of sound in the material.

The speed of sound is calculated based on the properties of the material. It is calculated according to:

$$c = \sqrt{\frac{\gamma E}{\rho}} \tag{3.45}$$

where L – smallest element extent, c – speed of sound in the material, E – Young's modulus, ρ – material density. γ – is calculated according to:

$$\gamma = \frac{1-\nu}{1-\nu-2\nu^2} \tag{3.46}$$

where ν – Poisson's ratio of the material.

The Young's modulus E is not constant for non-linear elastic materials and must therefore be set to a higher value. We use the approach used in Abaqus Explicit (Hibbit et al., 2007) whereby the critical time step is scaled down by a factor of $\sqrt{3}$ instead.

The strict requirements to the timestep value is the main disadvantage of the explicit time integration approach. It may lead to difficulties when analyzing "stiffer" materials. For instance, the analysis of a relatively soft pelvic floor with an average 100kPa Young's modulus and 0.45 Poisson ratio modulus and smallest element extent of approximately 1mm will give a critical time step of $5.13 \cdot 10^{-5}$ sec. On the contrary, a bone FE model with twice the smallest element extent and stiffness modulus of 3GPa will have a critical time step of $2.96 \cdot 10^{-7}$ sec. This requires 173.2 times more computation to arrive at a stable solution which

would make simulation rates much slower.

However, the major advantage of the explicit approach for time integration is the fact that no global matrix assembly is required. That is only the case if the mass matrix is diagonalized using mass clumping techniques as described in (Bathe, 1995).

3.3 Projection based Contact Method

3.3.1 Overview

During childbirth the fetus is at some stage in direct contact with the maternal uterus, uterine cervix and the birth canal. These interactions can be of crucial importance for the correct simulation of childbirth. For this purpose our simulation system includes a mechanical contact model specifically applicable to the types of contact occurring during childbirth.

In computational mechanical contact most of the phenomena involved must be assessed in a discrete fashion. This implies both *temporal* and *spatial* discretizations, whereby most phenomena cannot considered in a continuous manner. Time discretization requires to perform physical calculations at a finite time-step, whereas *spatial* discretization implies a limited precision at which geometry of an object can be described. Additionally, in the context of mechanical contact, the interaction of two surfaces can only be described at a given finite precision with the use of geometrical primitives, i.e. a node, 2D segment or 3D face. Section 3.3.3 is dedicated to describing details of the various discretization types used in the context of computational mechanical contact.

Contact problems typically involve two or more bodies undergoing partial surface or volume overlaps. There are simpler cases when the active contact surface is known a priori. For other contact problems where no pre-defined *active* contact zone is known it is necessary to perform global contact searches to find the overlapping surface sections. This process is an important part of computational contact and is called *contact detection*. *Contact detection* is the process of identifying whether the simulated objects are interacting through touch or impact. In computational contact it is often a process very similar to the process of *collision*

detection used in computer graphics and games applications. This is particularly true if the simulated objects are represented using polygonal meshes consisting of large numbers of vertices and triangles. The process of *collision detection* in its essence can be summarized as the process of finding out whether triangles from two given meshes are intersecting and if so, to identify those triangles. This is often done by performing pair-wise intersection tests of the triangles. In cases where the mesh complexity of the simulated objects is high, which implies large numbers of triangles that need to be tested for intersection, it is necessary to improve the efficiency with which the testing is performed. The details of the contact detection techniques utilized and the types of optimizations employed are discussed in Sec. 3.3.4.

As mentioned above, the fetus typically interacts directly with the soft tissues of the maternal birth canal but the soft tissues will deform and come into contact with the maternal pelvis that is enveloping the birth canal. This leads to indirect contact of the fetus with the maternal bony pelvis. Therefore, the simulation system should be able to model the contact of the fetal head and trunk with the bones of the maternal pelvis.

There is also an additional type of contact called self-contact whereby a single body is deformed to the extent that some parts intersect other parts of the same body. Whereas the self-contact problem is relevant to many biomechanical simulations (Johnsen et al., 2015), our approach to the simulation of contact during childbirth does not need to consider self-contact.

3.3.2 Contact with volumetric meshes

As discussed in Section 3.1.4, the FE meshes participating in the simulation are volumetric tetrahedral meshes, whereas the objects representing the rigid bodies are shell ("hollow") meshes consisting of triangles. The existing contact methods, however, focus on finding overlaps of only the outer shells or the surfaces of the meshes. The tetrahedral mesh is generated from a shell mesh input. The tetrahedrization method (TetGen described in Section 3.1.4) used has the option of preserving the surface topology, which was enabled, thus allowing us to reuse the same input surface mesh. However, the vertex indices comprising the triangles

are changed due to the fact that the vertex array has been changed.

The tetrahedral mesh generation method based on TetGen (Si and Hang, 2015) provides a way to map the face indices from the original surface to the generated volumetric representation. It is necessary to identify the vertex from the surface mesh corresponding to the one from the volumetric. This mapping process can be realized using a naive approach to correspondence mapping from the surface mesh to volume vertices. It would involve performing a linear search through all vertices of the the tetrahedral mesh for each vertex of the surface mesh ($O(N^2)$ operation). For instance, given a vertex from a surface mesh that was identified as a contact point, the corresponding node needs to be found in the volumetric mesh. To accelerate the mapping a bounding volume based optimization can be utilized, but the process is still costly.

In order to avoid the need for mapping altogether, we replace the original surface with the extracted surface of the volumetric mesh. Instead of searching for corresponding nodes between surface and tetrahedral meshes, the surface mesh is extracted from the tetrahedral mesh, by collecting all tetrahedra sides that belong only to a single tetrahedron. Such sides are the triangles of the surface. These triangles will contain the correct indices of the vertices in the volumetric mesh as well as the surface mesh. The correspondence in this is case is direct as the same indices are applicable to both meshes. Consider a surface triangle, consisting of vertices v_1, v_2 and v_3 with indices i_1, i_2 and i_3, has been found penetrating another object. These vertices need to be manipulated in the tetrahedral FE mesh in order to resolve the penetration. With the described approach the indices i_1, i_2 and i_3 from the surface mesh can be directly used to index into the FE mesh and access the correct nodes and manipulate them.

3.3.3 Contact discretizations

The Figure 3.19 show continuous bodies, but in numerical contact the surfaces are approximated by discrete elements. The surfaces in contact described above are typically represented as a collection of nodes and/or *segments* in the 2D case and as faces in the 3D case, as opposed to the continuous representation of surfaces in continuous mechanics of contact. Note that the term *segment* will be used

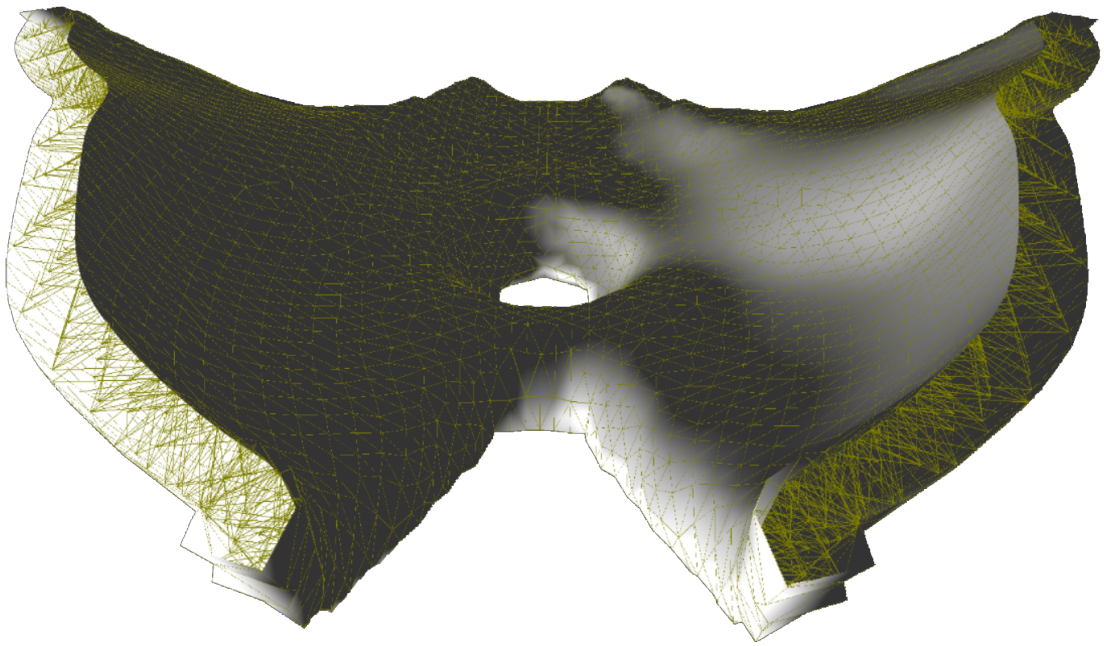


Figure 3.18: Frontal view of the volumetric tetrahedral mesh of the pelvic floor (yellow outlines) embedded into its surface mesh (white surface). The clipping is added to demonstrate the volumetric tetrahedra inside the surface mesh.

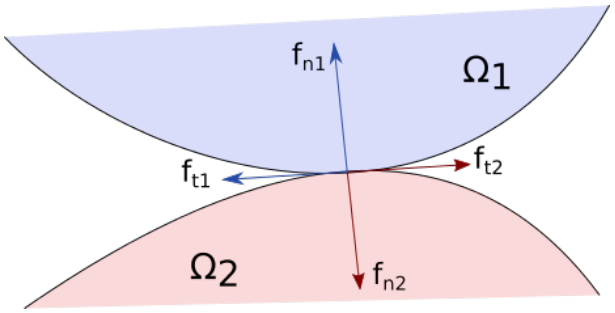


Figure 3.19: Two surfaces Ω_1 and Ω_2 of two bodies in contact. The normal components of the reaction force f_{n1} and f_{n_2} and the tangential components f_{t1} and f_{t2} for the first and the second bodies respectively are shown equal in magnitude and opposite in direction.

further to mean both a face and segment for the 3D and 2D cases respectively.

When contact detection is performed on a discrete mesh at discrete time-steps, it is often the case that time discretization errors occur. One of the most common errors is the occurrence of penetrations between contact bodies, which should not occur during contact between solid objects. Such errors are the result of the discrete approximation used. Figure 3.19 shows an example of contact pair where one of the slave nodes is penetrating the master. This penetration an example of a discretization error and should be resolved. The resolution of the penetration is a part of the *contact resolution* process, which is described in Section 3.3.5

The way the elements of the contact interface are treated during their interaction is determined by the choice of the contact surface discretization. These interacting elements are commonly assembled into *contact pairs*. The identification of the suitable *contact pairs* is the ultimate result of the contact detection stage. There are three common ways for the surfaces to be discretized (Yastrebov, 2013):

1. *node-to-node* (NTN)
2. *node-to-segment* (NTS)
3. *segment-to-segment* (STS)

Node-to-node

The *NTN* type is the simplest of discretization types and was initially described by Francavilla and Zienkiewicz (1975). This type of discretization considers the interaction of each of the nodes on the*active contact zone* of one of the bodies in contact to the corresponding nodes of the second body. An example of a 2D contact with NTN discretization is shown in Fig. 3.20. For the NTN discretization to be applicable, the surfaces in contact must have conformance. The conformance is achieved by the one-to-one correspondence of the nodes on the *contact interface*, whereby each of the nodes on a surface has a unique corresponding node on the opposing surface in close proximity. This conformance leads to good performance of the discretization for purely normal interaction, where no tangential slide is

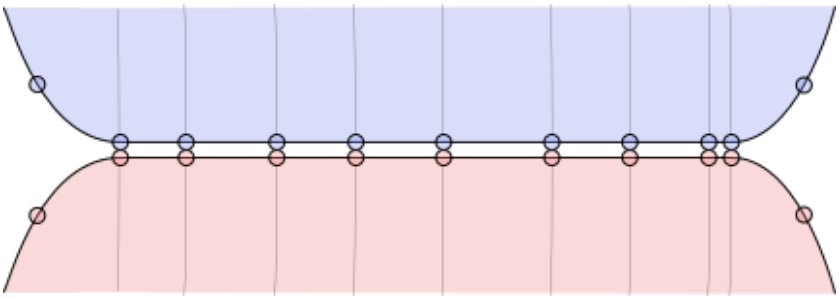


Figure 3.20: Node-to-node (NTN) discretized contact. Each node from either surface has a corresponding node on the other surface signifying the inter-surface conformance.

not present. The presence of slip will lead to a non-conformant configuration where some nodes no longer have a corresponding node.

The *NTN* discretization that allows only infinitely small slip requires less complex contact detection as compared to the other types of discretization. It is only required to find the nodal contact pairs by using simple proximity queries. For each node on one surface the queries are performed to identify the closest point on the second surface. Note that without slip the *contact pairs* are stable and persist throughout the simulation. In the presence of slip in the form of tangential motion, the nodes forming contact pairs that originally were in close proximity will move away from each other. This will brake the contact pairs and the inter-surface conformance.

Node-to-segment

The *NTS* discretization considers the interaction of the nodes from one of the two contact surfaces and asegment from the other surface, both constituting a *contact pair*. The initial work on NTS based contact mechanics is presented by Hughes et al. (1977) This type of segmentation is considered to be more complex in implementation but at the same time presents a more versatile approach (Yastrebov, 2013). The versatility of the approach is ensured by the ability to treat large deformations in contacting surfaces and large tangential sliding of the contact interface. A moderate mismatch in mesh resolution between surfacesispermitted. However, the mismatch is only allowed in case correct assignment of which sur-

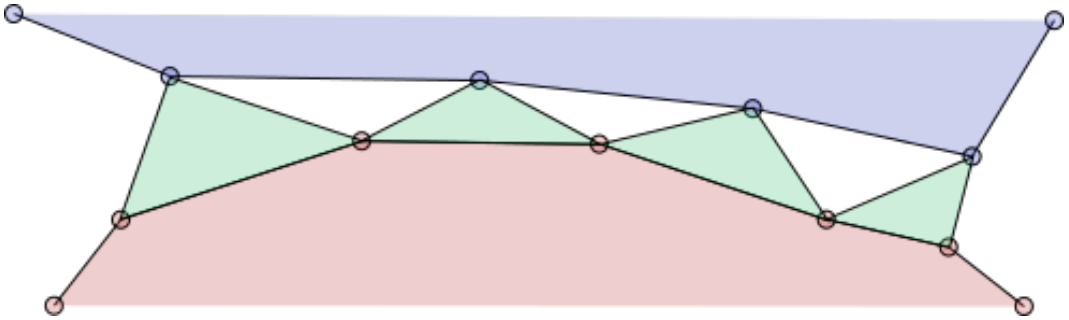


Figure 3.21: Node to segment (NTS) discretized contact. Contact pairs (green) showing a node from *slave* (blue) surface and a corresponding segment from the *master* surface (red).

face's nodes are used and which surface's segments are used for contact pairs. The specifics of this assignment are discussed further. The approach, however, has problems with stability and robustness in edge cases (Zavarise and Lorenzis, 2009), whereby some segments will spuriously penetrate each other. An example of such an edge case is shown in Figure 3.22, when the slave surface has a coarser mesh leading to issues with spurious penetrations. Figure 3.21 demonstrates how the *NTS* discretization can be depicted for a 2D contact between two surfaces.

The concepts of *slave* and *master* surfaces are particularly applicable for the case of NTS discretization. As mentioned, the choice of surface from which the nodes are to be used and vice-versa is important. Commonly, the surface with finer subdivision is assigned the role of the *slave* surface whereas a more coarse surface has the role of the *master* surface. In the context of NTS discretization it is the slave surface's nodes that are checked against penetration with the master surface's segments.

The reason why the *slave* surface is chosen to be the finer one is that the node based discretization ignores surface's continuity. The effects of swapping the slave/master assignment from the correct to the incorrect way are shown in Fig. 3.22. Note how the inverted slave/master surface assignment fails to prevent penetrations even in a simple scenario. These issues are mainly manifested in the case when the fine master surface has considerable curvature or protruding features.

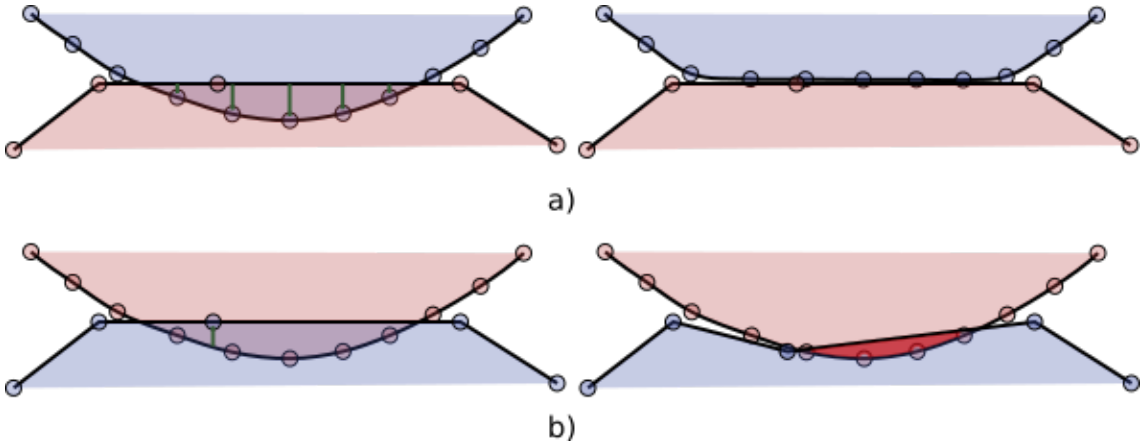


Figure 3.22: a) Correct choice of *slave*/*master* assignment allows for correct detection (green lines showing detected penetrations) and resolution of contact (right).
b) Incorrect choice of *slave*/*master* surface assignment can lead to undetected spurious penetrations (shown in bright red).

The contact detection stage is performed to identify contact pairs of nodes and segments. This process is comparatively complex when compared to the contact detection in the *NTN* discretized case. The process is described in detail in Section 3.3.4, but it can be summarized as a global search of intersecting surface triangles. The slave surface nodes are then checked against the master surface segments (faces) and the contact pairs are established. Apart from the contact pair specifying the node and the corresponding segment, in case of an *NTS* discretized representation of the body in contact, it is necessary to identify all the penetrating nodes along with the following information (depicted in Fig. 3.23):

- Depth of the penetration
- Point on the master surface segment where the penetration initially occurred
- Normal of the penetrated master segment

The *NTS* discretizations allow large slip and large deformations which leads to invalidation of some of the contact pairs. Nodes experiencing large tangential

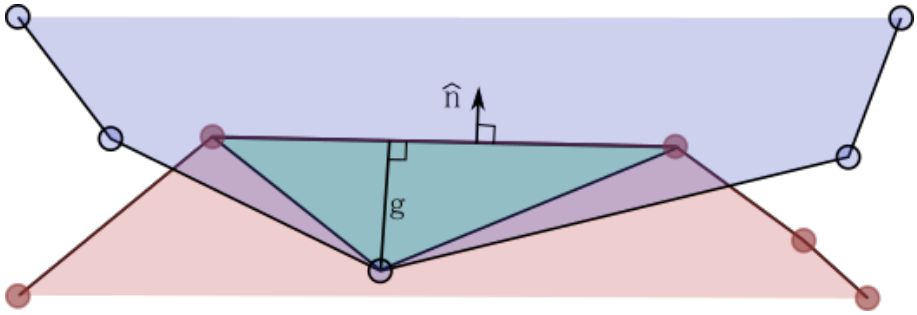


Figure 3.23: A *node-to-segment* discretized contact interface showing a contact pair (green). The depth of penetration is g. The normal of the penetrated master segment is $\hat{n}$, also known as the *gap*.

slips may becomes separated enough to break the contact pair. This implies that contact detection needs to be performed every time a contact pair has been invalidated. Additional contact detection leads to additional computational cost which can be counted as a disadvantage of the method.

Segment-to-segment

The *STS* discretization is considered to be the most complicated type (Yastrebov, 2013) whereby contact pairs are created by assigning each segment from one surface to the corresponding segment of the other surface. The introduction of this type of discretization was done by Simo et al. (1985).

3.3.4 Contact detection

Types of contact

The simulation of childbirth presented in this thesis features three types of objects:

1. Static rigid bodies – bodies that are not movableor deformable but have collision enabled when other bodies interact with them. An example of such object in the specific case of childbirth simulation is the maternal bony pelvis.

2. Dynamic rigid bodies – bodies that behave like classical rigid bodies and also interact with other simulation objects. Examples are the fetal skull and the fetal trunk that are able to move as an effect from external forces acting on them.

3. Deformable bodies – bodies that are modelled as deformable FE meshes. They can deform based on the boundary conditions imposed on them and the body forces applied. These bodies can interact with both of the rigid body types through a contact method.

Note that the static rigid bodies are marked as interacting with other bodies of the same type but under normal circumstances they may never come into contact unless they start out in an intersecting state.

Bounding Volume Hierarchies

The meshes used in medical simulations typically consist of large numbers of triangles representing the surface of the simulated object. A naïve approach to contact detection involves testing each pair of triangles for intersections. The computational complexity of this process will be $O(n^2)$ (Harris et al., 2007), which, in case of models with a high polygon count n, will yield reduced performance and correspondingly, low frame rates. One of the most popular solutions to this problem is spatial partitioning using Bounding Volumes (BVs). This approach divides collision detection into two consecutive phases. The firstphase, often called the *broad-phase*, involves pruning of the more distant (unlikely to collide) polygons from the list of all polygons to undergo the intersection tests, leading to reduced complexity $O(n\ log\ n)$. After a small number of potentially colliding polygons areidentified, the *narrow-phase* collision detection takes place. This phase involves pairwise intersection tests on the set of potentially intersecting triangles.

Bounding volumes allow ruling out distinctly separated objects without any primitive tests but as soon as an overlap of two BV's is found, all of the primitive pairs need to be tested. For the purpose of optimizing contact detection further it is beneficial to use bounding-volume hierarchies (BVHs). A BVH is typically

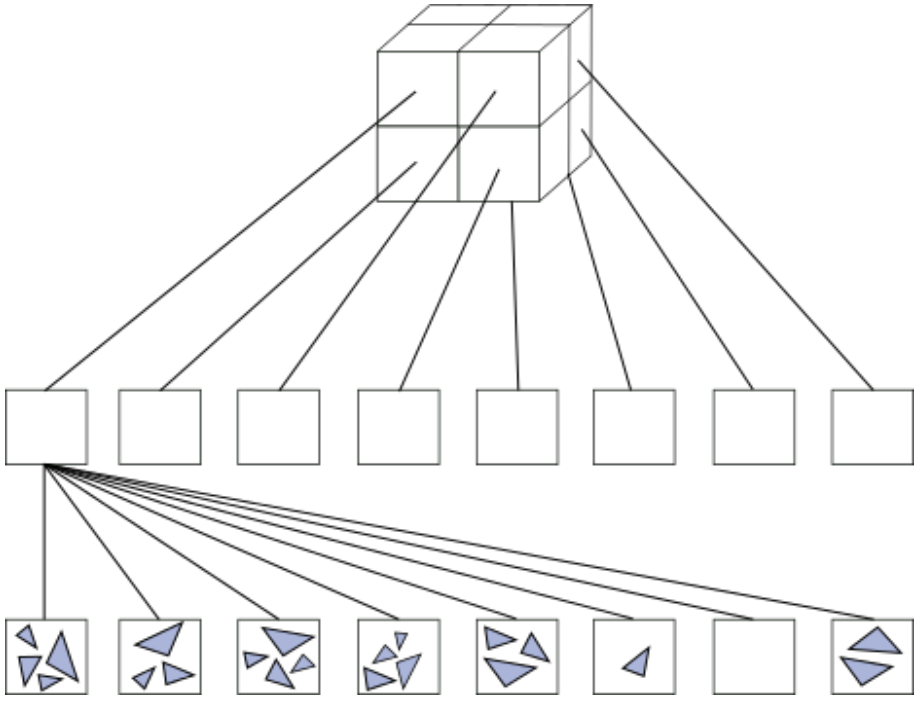


Figure 3.24: An octree BVH. Root node at the top shown as a 3D AABB with subdivisions for child nodes. The child nodes are shown lower down. The lowest level of *leaf* nodes contains surface triangles.

represented by a tree structure with each tree node beingrepresented by a BV along with all the surface primitives enclosed by that BV. A basic representation of a BVH tree is shown in Figure 3.24. Each node can contain a number of other nodes which are called *child nodes*. The combination of all children of children are the *descendants* of a node. Each tree must a have a *root* node, which serves as the topmost container enclosing all its *descendants*. A node with no children is called a *leaf* node. Typically only leaf nodes are assigned surface primitives. The nodes of BVH tree are commonly represented by axis-aligned bounding boxes (AABB's). These boxes will always be oriented along the global coordinate axes and will always enclose all the child nodes of a node and all the primitives contained in it.

The maximum number of children for each node determines the *arity* of the tree. A tree of *arity* 2 has two nodes for each non-*leaf* node and is called a *bitree*. A tree of *arity* 8 is called an *octree*. Two examples of BVH trees are shown in

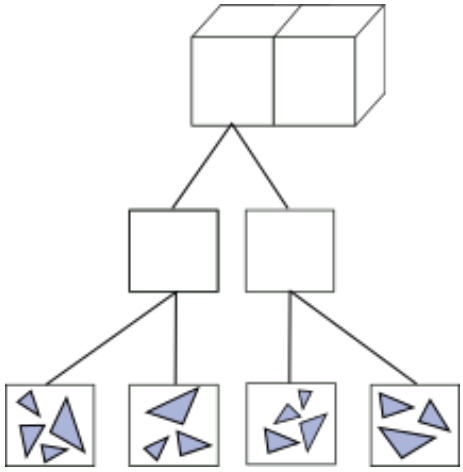


Figure 3.25: A bitree BVH. Root node at the top shown as a 3D AABB with subdivisions for child nodes. The child nodes are shown lower down. The lowest level of *leaf* nodes contains surface triangles.

Figures 3.25 and 3.24.

Tree representation In case of a tree based BVH, there are two common ways of encoding the hierarchy representation (Gargantini, 1982). The difference between the two methods is in the way the hierarchy is stored and laidout in memory and also the way the relationship between nodes is specified. Note that both of these representations are equally applicable to bitrees and octrees.

The first type is the explicit *pointer based tree* representation in which all the nodes (leaf nodes and non-leaf nodes) are stored explicitly. The relationships between nodes, such as parent-child relationships are also denoted explicitly through using pointers or references. The root node has no parent and has 8 child nodes (in case of an octree) or 2 child nodes (in case of a bitree), stored as an array of pointers to each child object. The relationship arrows from Fig. 3.24 and Fig. 3.25 can be directly interpreted as pointers or references. The nodes that have zero pointers to children are *leaf nodes* and store an array of triangle indices that are enclosed by it.

The second type of representation is called a *linear tree* (Gargantini, 1982) whereby only the leafs of the tree are explicitly stored in sequential manner.

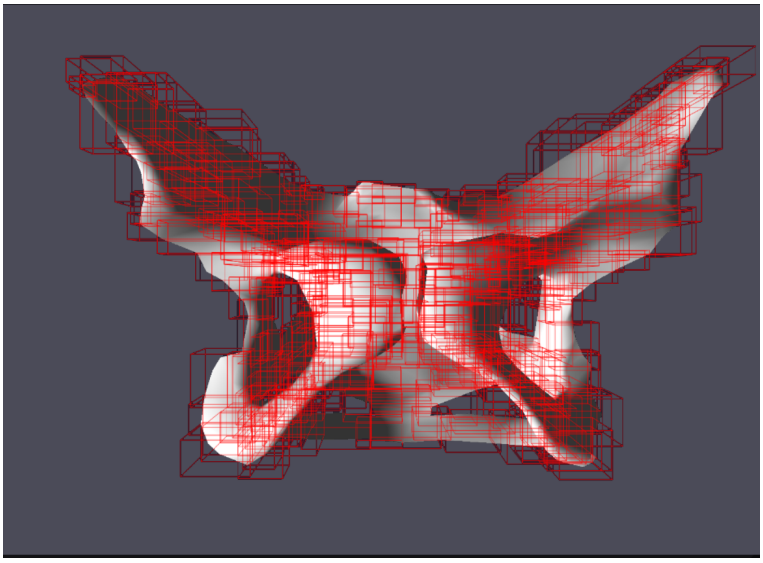


Figure 3.26: Octree built around the model of the maternal pelvis.

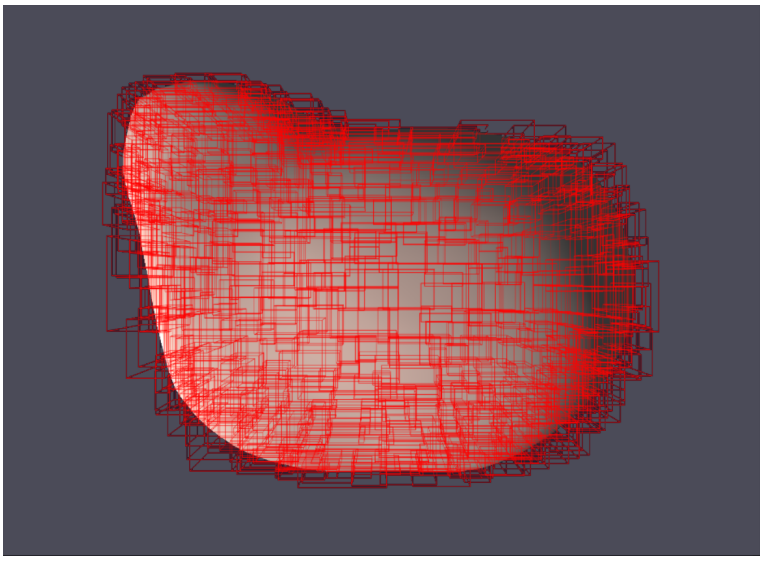


Figure 3.27: Octree built around the model of the fetal head

This type of tree is also called *pointer-less* because indirect memory access (using pointers or references) is not used to find the child nodes or the parent node of a given node. Instead of pointer based access, the relationship between nodes can be derived implicitly using a function. The function allows to find the index of the first child of the current node and also deduce the index of its parent. The principle used for this type of indexing is referred to as a *Morton code* or *z-order curve* ordering(Morton, 1966). The technique allows reducing the dimensionality of the key from 3 to 1, leading to linear indexing. Further addition of a one-dimensional data structure, such as a binary search-tree or B-tree, allows for efficient tree search and traversal.

Tree construction The process of constructing a BVH tree can be approached in two ways which are the *top-down* and *bottom-up* approaches. A given surface mesh is comprised of a large number of faces and the tree construction process is essentially dedicated to distributing the surfaces' faces into leaf nodes in the most efficient way possible. The efficiency is both for the sake of improved construction performance and even more critically for the performance of the subsequent traversal and collision detection.

In case of the *bottom-up* approach the algorithm can be summarized as follows:

Algorithm 1 Algorithm for the *bottom-up* BVH construction

1: **for all** face in mesh **do**
2: Enclose face into a bounding box thus creating a pool of leaf nodes
3: **end for**
4: **for all** node from pool of leaf nodes **do**
5: Find $N-1$ closest nodes, where N - tree's arity
6: Build a new parent node around the closest nodes and the seed node
7: Repeat until a single *root node* is left that is enclosing all faces
8: **end for**

The approach is named *bottom-up* because the construction begins at the lowest level of the tree. As seen on Fig. 3.28 the leaf nodes (bottom of the tree) are constructed first around each of the surface triangles. The figure demonstrates further steps required to build a binary tree around a mesh which involves combining lower level node pairs into new nodes until a single root node is created.

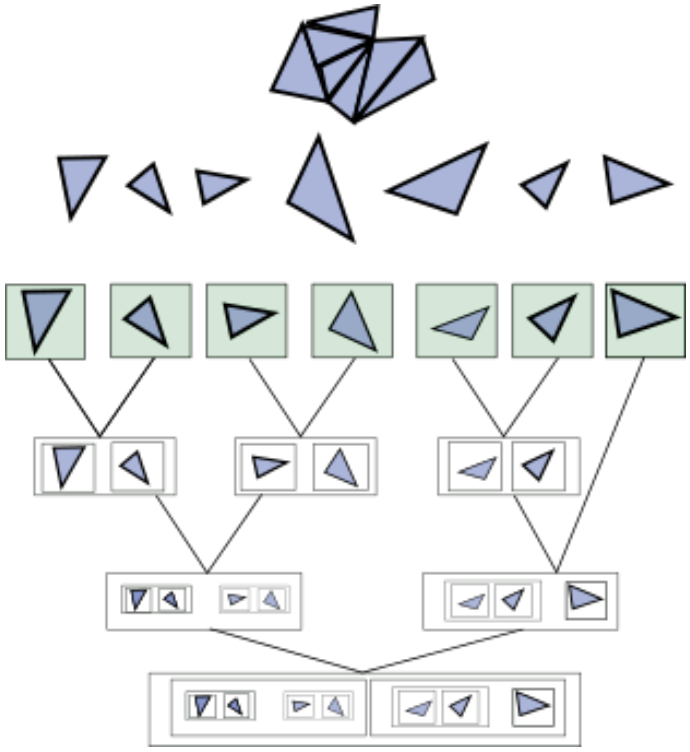


Figure 3.28: A bottom-up construction of a bitree.

The same approach is directly applicable to an octree with slight variation. Each further step in the construction ascends towards the root of the tree.

Although this type of tree construction is applicable to octrees (Franklin and Akman, 1985) using a stacking approach, it presents more ambiguity in determining the suitable $N-1=7$ nearest nodes. The stacking approach is effective, but presents additional computational load. The ambiguity is reduced in case of a bitree construction as the task of identifying the $N-1$ closest nodes is reduced to finding a single nearest node.

This approach will produce a tree with each face having a dedicated tree node which is not always desirable as it may be detrimental to collision detection performance. This issue can be addressed simply using tree pruning after the construction. The tree nodes containing too few face primitives can be removed and the primitives can be transferred to the parent nodes.

The *top-down* approach starts at the top of the hierarchy tree by constructing

the *root node* first around the whole of the surface mesh. Then it progresses further down the tree by subdividing each node according to a given *heuristic*. The heuristic is there to determine if a node should be subdivided in order to provide optimal performance. The algorithm can be summarized as follows:

Algorithm 2 Algorithm for the *top-down* BVH construction

```
1: Build a single root node and add all faces to it
2: Refit node's bounding box around all faces in current node
3: if heuristic shows the current node should be split then
4:    Create N child nodes and assign parent/child relationships
5:    for all face in current node do
6:       Determine which child node should the face be transferred to
7:       Transfer face to appropriate child and remove from current node
8:    end for
9:    for all child in children do
10:      Go to 3
11:   end for
12: end if
```

Tree traversal The main aspect of the broad-phase collision detection using a BVH is the traversal of the hierarchy tree. The objects modelled in the simulation system are allowed to undertake arbitrary rotations. This means that octree node AABB's of one object can be rotated with respect to the octree node AABB's of the other. Then during contact detection these relative orientations turn the AABB's into oriented bounding boxes (OBB's).

The orientations of the tested OBB's are usually represented by matrices as suggested in (Ericson, 2004). One way to perform the OBB intersection tests is by modifying the extent vectors of each of the boxes by their orientation matrix. However, one of the vector by matrix transformations can be avoided if the tests are performed locally to one of the OBB's. This can be achieved by calculating the transformation matrix that represents the first OBB in the coordinate frame of the second and performing the transformation only on the second. Fig. 3.30 and 3.31 illustrate the idea. Forcases where the BVH consists of many OBB's this can improve the performance. Equation 3.47 can be used to calculate the localization matrix.

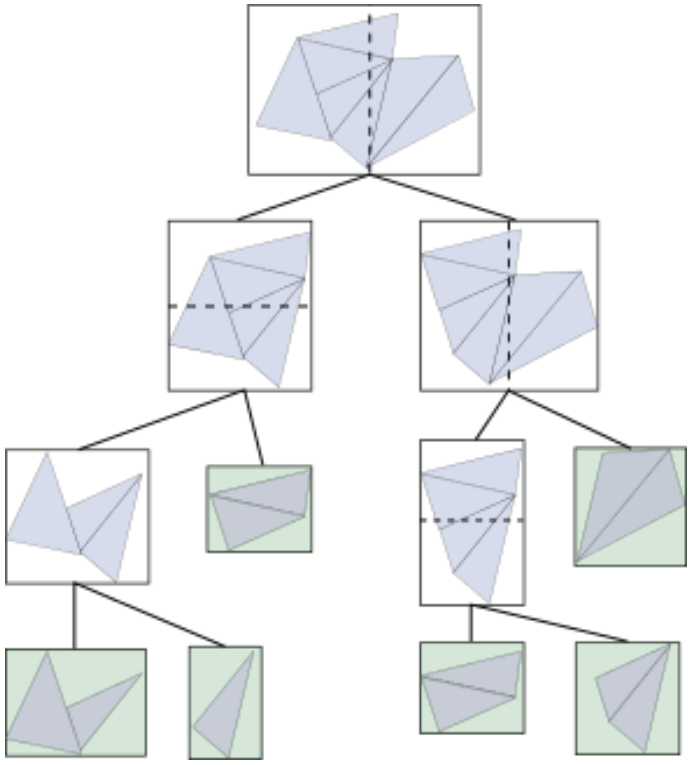


Figure 3.29: A top-down construction of BVH tree. The *root* node is constructed first and further subdivided until termination based on a heuristic. The heuristic here is based on the maximum number of triangles per leaf node (green), which is 2.

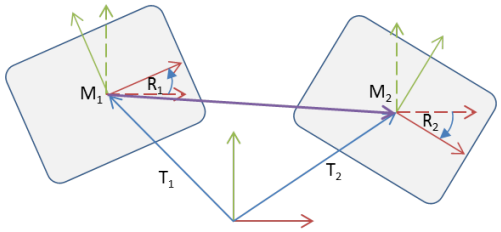


Figure 3.30: Global and local coordinate frames of OBB's are shown with dotted and solid arrows respectively.

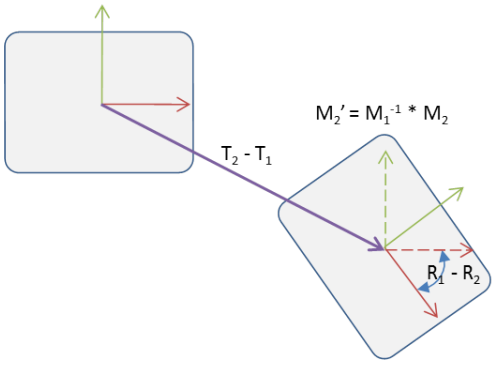


Figure 3.31: Representing the right OBB in the coordinate frame local to the left.

$$M_2' = M_1^{-1} \cdot M_2, \tag{3.47}$$

where M_2 - *from* matrix, M_1 - *to* matrix, M_2' - localization matrix from M_2 to M_1.

Finding contact surface

The narrow phase of our contact detection approach is divided into the following stages:

1. Finding a contact border
2. Node penetration point and depth identification with ray-casting
3. Utilizing temporal-coherence to maintain the contact surface

The *narrow* phase of collision detection involves determining the exact intersections between the potentially intersecting faces identified during the *broad* phase. The generally accepted approach for testing polygon-polygon intersection uses the Separating Axis Theorem (SAT) (Miller, 1997). The SAT states that, if two polygonal objects are not colliding, there exists an axis that separates the projections of the objects onto that axis. Thus, if an axis on which the projections of the objects do not overlap is found, the objects are not intersecting. In the 3D case, however, there are infinitely large number arbitrary axes that could be tested, which is not feasible. The important point here is that for SAT the number of test axes is fixed and the number of required tests is calculated as in Eq. 3.48.

$$N_t = f_1 + f_2 + e_1 \cdot e_2, \tag{3.48}$$

where N_t - number of required tests, f_n - number of faces of the n-th object, e_n - number of edges in the n-th object.

For triangles the number of axes is 2 face normals and 9 edge vectors and their cross products.

There are several implementations of the SAT for determining intersection between two triangles (Miller, 1997). Each of the implementations vary in terms of

the computational complexity and robustness. The algorithm with least computational cost and best robustness is required for our purpose. The one presented in (Miller, 1997) presents a suitable solution in the context of intersection testing for large numbers of potentially intersecting triangle pairs.

The SAT based approach to contact detection utilized for our contact method only allows finding the triangles that are currently intersecting other triangles, as opposed to the methods that allow detecting triangles enclosed by the volume of the other mesh. The result of such a method for two convex objects is a list of triangles located at the intersection curve. These triangles form the border around the contact surface. An example of a *contact border* for a 2D case is depicted in Fig. 3.32 by the green segments. The border in the 2D case is disjoint and consists of the segments of the slave surface currently intersecting the segments of the opposing master surface. For a 3D case the contact border is typically a closed contour made of a band of triangles currently intersecting the master surface as seen on Fig. 3.33. In the figure the contour triangles are shown in green and the contact surface is shown in red. Note that the number of contact surfaces can vary arbitrarily based on the surface curvature and complexity.

The main issue at this stage is that the segments or faces that are currently penetrating the master surface cannot be similarly detected using SAT. They are internal to the master object's volume, but they are not intersecting any of the master triangles. For the purpose of identifying such segments and faces we utilize ray-casting and temporal-coherence based caching, discussed further. With these methods applied the full contact surface, which is the combination of the green and red segments (triangles) in Figures 3.32 and 3.33.

Ray-casting to find penetration depth

The process of BVH traversal with consequent triangle pair tests identifies the list of intersecting triangles. The nodes comprising the intersecting triangles are then used to perform ray-casting to find if the nodes are penetrating the master surface.

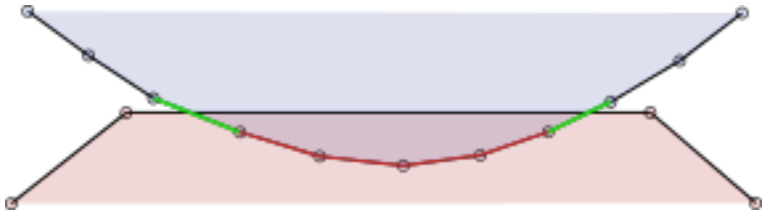


Figure 3.32: Contact border for a 2D case of a circular object (blue), which is the slave, penetrating a trapezoid (red) master. The border is disjoint and consists of the two segments intersecting the master surface (green). Segments fully penetrating the master surface (red) are not detected

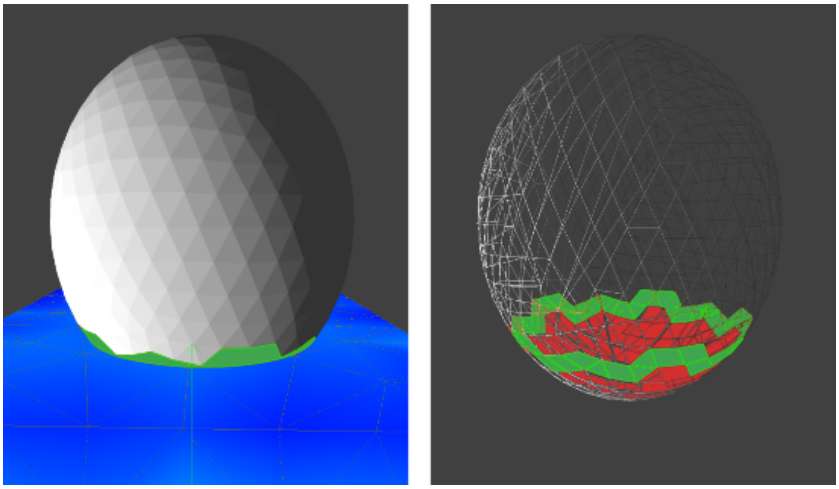


Figure 3.33: Contact border for a 3D case of a spherical object (grey) penetrating a cube (blue). The contour is continuous and consists of all the triangles intersecting the master surface (green). Triangles fully penetrating master are inside the master surface are not detected as they are not intersecting any triangles of the master surface (shown in red).

The equation for the ray is

$$\vec{R_i} = \vec{P_i} + \vec{R_d} \cdot t \tag{3.49}$$

where $\vec{R_i}$ - point on the ray, $\vec{P_i}$ - origin of the ray (slave node position), $\vec{R_d}$ - ray direction (master face normal), t - ray parameter.

The technique (Möller and Trumbore, 2005) for ray vs triangle intersection test allows identifying the parameter t of the ray at which the intersection occurred. To identify the particular master triangle that is penetrated by the slave node, the ray originating from the node and directed anti-normally is cast. The ray is tested with the master triangles. During ray-casting the position of the slave node P_i is used as the ray origin O_o.

The larger number of master triangles with which the slave node ray needs to be tested requires some optimization of process. A BVH based approach is found to be an effective way of doing so. The BVH based optimization procedure for the ray-casting is similar to the one used for identifying overlapping triangles. The algorithm can be summarised as follows:

Algorithm 3 Algorithm for the ray-casting approach to find nodal penetrations

1: Recursively traverse the octree while identifying the non-empty leaf octree nodes
2: Perform Ray-AABB intersection tests on each of the bounding boxes of the octree and store intersection information in array
3: Sort the list of identified leaf nodes according to the distance of the ray's intersection point to the origin
4: **for all** node in sorted leaf nodes **do**
5: **for all** primitives stored in each leaf node **do**
6: Perform a ray-cast against all the primitives stored in each node
7: **if** ray intersection is found **then**
8: store in the list of ray intersections
9: **end if**
10: **end for**
11: **if** the list of ray intersections is not empty **then**
12: break loop 4
13: **end if**
14: **end for**

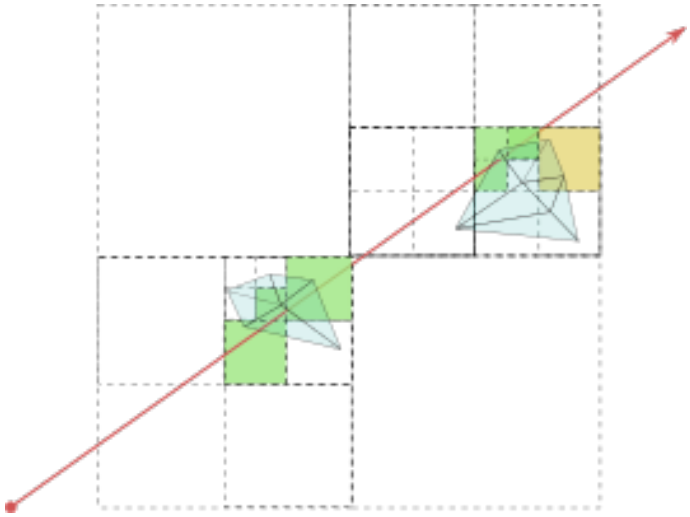


Figure 3.34: Figure demonstrating octree node intersections and primitives inside them.

The initial stage of ray-casting is only focused on identifying the list of nodes intersected by the ray. All the nodes intersected are collected and sorted based on the distance t . That is the nodes are arranged in the order in which they were hit by the ray as it travelled along its direction.

The sorting allows an early escape from the outer loop 4. It can then be assumed that the t distance (Fig. 3.35) octree node has the lowest value as compared to all the potential ray intersection distances with the primitives inside the octree node. Therefore, if the closest octree leaf contains primitives intersected by the ray, it can be assumed that the rest of the leafs need not be visited.

The same principle according to which only the first node hit by the ray needs to be processed cannot be applied to the primitive tests within a node. The first intersection with a primitive is not guaranteed to have the closest intersection distance. Only after all primitives within the node are tested and all intersection distances are known, then the closest intersection can be found.

Ray-casting in degenerate cases

A contact region may often persist throughout the duration of several updates. When a projection based contact method is used, the slave nodes from the region

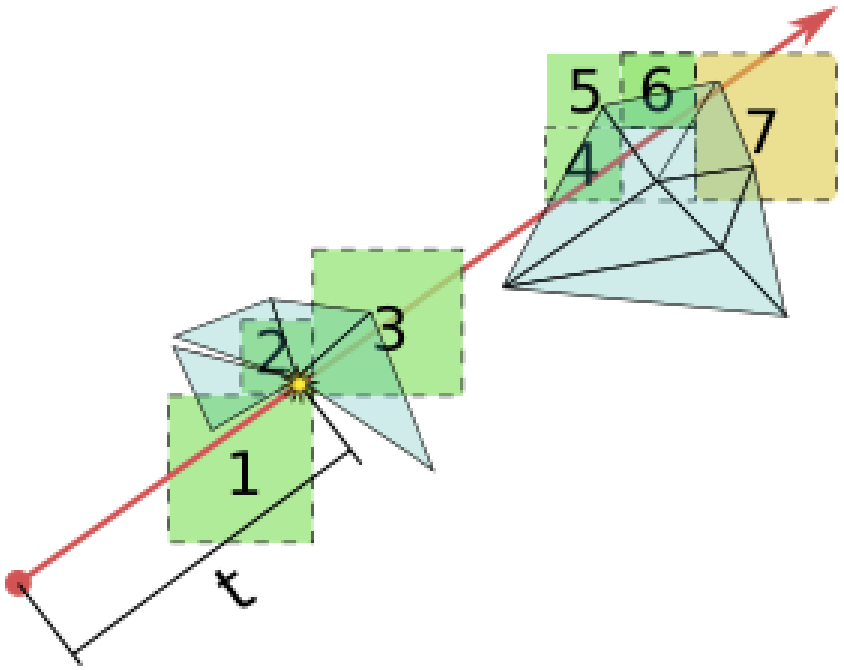


Figure 3.35: Ray intersection in the closest bounding box precludes any intersections in the further nodes. The intersection in the node 2 will occlude the rest of the triangle intersections.

will be *projected* from within the surface to remained exactly on the corresponding master surface faces. During the consequent iterations a ray-cast will used again to determine if these *projected* nodes have moved back into penetration.

The top part of Figure 3.36 demonstrates the *projected* nodes located directly on the master surface. During ray-casting the position of the slave node P_i is used as the ray origin O_o. For the case of the slave surface's *projected* nodes described above, the origin of the ray will be directly on the master surface. The ray pointing outwards along the master surface normal may arbitrarily fail to intersect the surface due to the numerical floating point errors. The ray-triangle intersection test may or may not consider the ray with an origin laying directly on the surface of the tested triangle as laying fully on one side of the triangle without intersecting it. This is an example of a degenerate case for the ray-triangle intersection test.

This issue is virtually non-existent when using the penalty force based contact methods. In the cases of these methods the penetrations are never resolved completely, leaving the slave nodes always in the state of slight penetration. The penetration depth is typically considerably more than the scales where the floating point errors are considerable.

We are using a projection based contact method where the penetrations are resolved perfectly which in turn has a trade-off of the issue described above. We address this issue by enabling a "run-up" margin M for the ray. This implies shifting the origin of the ray in direction opposite to its direction thus forming a run-up region. The equation for the ray then becomes:

$$\vec{R_i} = \vec{P_i} + \vec{R_d} \cdot (t - M) \tag{3.50}$$

where $\vec{R_i}$ - point on the ray, $\vec{P_i}$ - origin of the ray (slave node position), R_d - ray direction (master face normal), t - ray parameter, M - ray origin "run-up" shift margin.

When such a technique is used, the ray-cast for the slave node residing on the master surface will never be missed due to floating point errors. The origin of the ray still may end up directly on the master surface, but the floating point error will be irrelevant as such a node will always be non-penetrating.

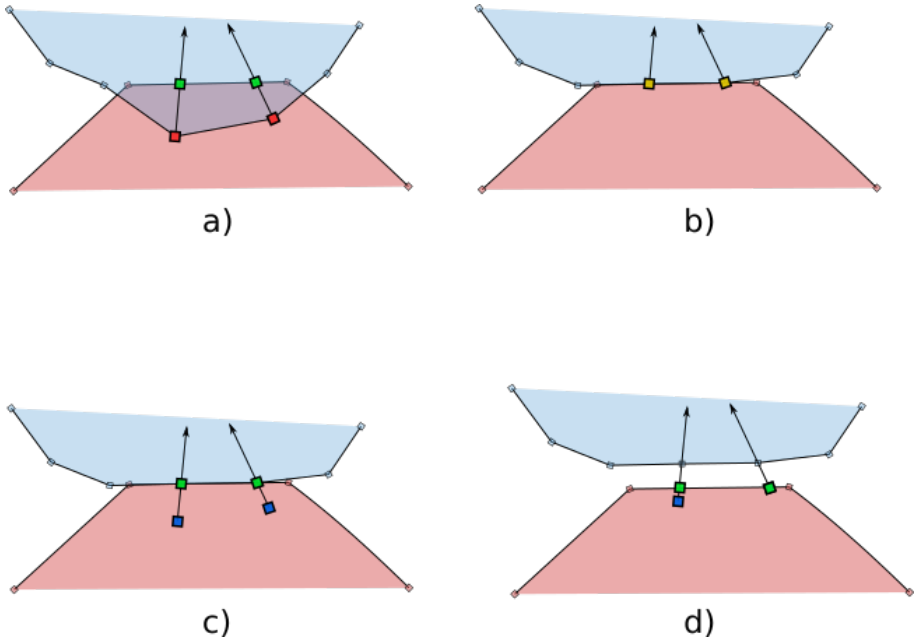


Figure 3.36: The "run-up" apprach for ray-casting in degenerate cases. a) penetrating nodes (red) are found and ray-casts are performed to find the projected locations (green). b) the nodes are projected onto the master surface (yellow). c) the projected nodes use a *run-up* for ray-casting to find if they penetrated again. d) a non-contact case where *run-up* ray-casting allows detecting if master surface is in the vicinity of the nodes.

An additional advantage of this technique is that it allows identifying the distance of the node to the master surface in the cases where the node does not penetrate it. This allows detecting if the node is in the vicinity of the master surface. The penetration value for such an approaching node is negative and the magnitude of the value gives the distance. A visual representation of the described case is presented in Figure 3.36 (d).

Utilizing temporal coherence to find the contact surface

This contact surface construction method is explicit in the sense that the full contact manifold at time t is calculated step-by-step by building up the list of surface triangles from the initial triangle that came into contact with the master surface first.

The illustration of this method is best shown by a series of snapshots of the contact area as the penetration is happening in Figure 3.37. As the slave descends towards the master surface, additional slave nodes and segments penetrate it. The previously intersecting segments are no longer intersecting the master surface, but they are cached. The newly intersecting segments are added to the cache list. Note that when larger steps are used, which means the slave mesh moves more between intersection tests, some of the segments will never intersect the master surface (shown in red in figure). Rather they will immediately "jump" past the master surface into the inner volume of the master object. These can be avoided by keeping timesteps and the velocities of the objects small. Additionally, if only the penetrating nodes (shown as blue circles) are of interest, which is the case with the projection based contact we are utilizing, the skipping a single segment presents no issue as the previous and next segments are sharing the nodes. The greed circles indicate the *potentially penetrating nodes* as they are shared by the intersecting segments. These nodes will be used during the ray-casting stage, but no penetration will be found. These are excluded from the cached list of penetrating nodes, thus the list only contains nodes that are known to penetrate the master surface. This illustration of the approach in 2D is readily applicable to the 3D case.

Octree refitting

The simulated meshes are deformable and can undergo large deformations during simulation. In particular, our simulation features a pelvic floor model that needs to stretch to accommodate the fetal head, thus changing its shape considerably. The octree that is built around the initially undeformed mesh needs to be rearranged and reshaped to match the new deformed shape. This process is called refitting (Tu and Yu, 2009). The quality of fit of the new octree needs to be considered as well while performing refitting in order to maintain the beneficial qualities of the BVH.

Various refitting techniques have been proposed to optimize the process such as lazy refitting and heuristic refitting (Larsson and Akenine-Möller, 2006).

We perform naïve refitting with persistent topology. The approach is naïve

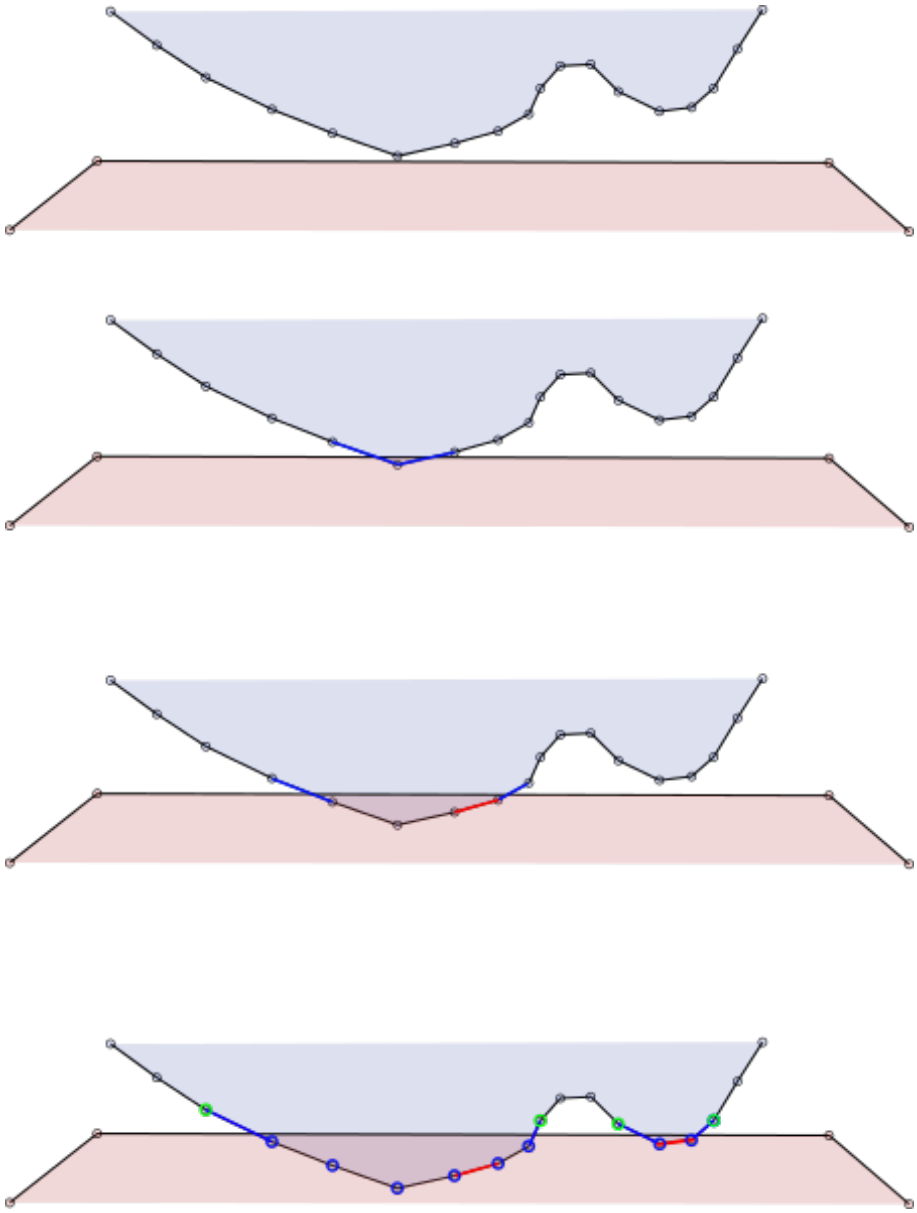


Figure 3.37: A step-by-step illustration of contact surface build-up based on temporal coherence. Time is going downwards. The list of penetrating nodes (blue circles) is gradually built up by keeping track of the previously penetrating nodes. The red segments are skipped due to large timestep.

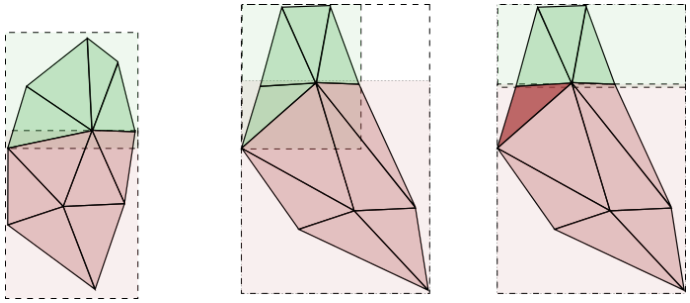


Figure 3.38: An undeformed mesh with a BVH formed around it (a). After deformation the BVH is naively refit without changing the topology (around the same triangles) resulting in larger overlap (b). A re-topologizing refit is performed and the dark red triangle is transferred to the lower bounding volume. Overlap is minimal.

in the sense that we do not employ any heuristics in the determination of which octree nodes should be refitted. We refit all tree nodes regardless of deformation. The refitting process is demonstrated in Fig. 3.38. Note that the nodes before and after refitting contain the same surface primitives, thus maintaining constant topology. The naive refitting results in larger overlapping areas as seen on the same figure. These overlapping areas are wasteful as they increase the number of primitive tests needed to check for contact. The costs of re-topologizing refit are relatively large as parts of the whole BVH tree need to be rebuilt and repopulated, while the costs of the overlapping bounding volumes is thought to be relatively small. Therefore, we only use the native approach to refitting.

3.3.5 Contact resolution

The contact method used in this work relies on treating contact stresses as internal stresses of a single continuum body consisting of the two interacting bodies. Notably, texts in solid mechanics call the internal traction forces "contact forces" (Holzapfel, 2000) as they can be interpreted as the forces occurring between two "virtual" surfaces in contact. In our case of real contact, the surfaces are distinct and real but the contact forces can be calculated in the same way as the internal traction forces are.

An adaptation of a figure from Holzapfel (2000) demonstrates (Fig. 3.39) how

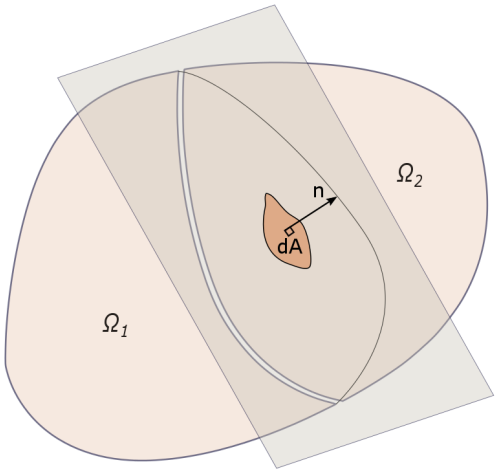


Figure 3.39: Two objects Ω_1 and Ω_2 in contact. Highlighted is an infinitesimal contact area dA with normal $\boldsymbol{n}$. Alternatively, interpreted as a continuous body experiencing internal stress in a plane with normal $\boldsymbol{n}$. Note that the separation is abstract and for demonstration only.

two bodies in contact can be interpreted as a single continuous body experiencing internal stress in a plane with a given normal. Most importantly, this internal stress can be integrated over the contact area dA to obtain the precise contact force acting on both bodies in contact.

Penalty and Lagrange Based Methods

As discussed above, presence of penetrations in contact is unrealistic in the sense that solid bodies do not inter-penetrate (apart from diffusion on microscopic level). Additionally, introduction of penalty forces or Lagrangian multipliers artificially modifies the assembly. In particular, the penalty forces are equivalent to the effect of a number of 1D bar or spring contact elements attached to the penetrating nodes (Kikuchi and Oden, 1988). These contact elements typically have linear, quadratic or exponential response, which is often in disagreement with the material response of the modelled body. Additionally, the material response of the simulated bodies varies from material to material whereas the choice of the contact element response remains the same.

One of the drawbacks of using a method, similar to the one used by Heinstein

et al. (2000) and Johnsen et al. (2015), is its sensitivity to the length of update time step. These methods can suffer from such sensitivity which can manifest itself as an unresolved erroneous penetration of slave nodes into the master surface. The size of the spurious penetration grows quadratically as the time step length is increased. The penetration only occurs when there is a force acting on the slave nodes in the direction towards the master surface. This relationship can be derived from the gravity force equilibrium. Note that this force is not necessarily gravity but can be any other external or body force acting on the nodes such as the uterine expulsion force.

According to Heinstein et al. (2000) a gap based contact reaction force can be calculated using:

$$F_p = g\frac{m_i}{\Delta t^2} \tag{3.51}$$

where $\boldsymbol{F_p}$ - penalty based reaction force, g - gap, m_i - nodal mass, Δt - update timestep.

Figure 3.40 shows a basic setup where spurious penetration can occur. The node at position p_t is penetrating the master surface by *gap* g. The penetrating node will move up towards the surface while the reaction force F is larger than force G pushing it into penetration. This is equivalent to *gap* g decreasing until it becomes equal to ϵ. At the equilibrium point where the forces are equal the movement will stop subject to minor oscillations. The derivation for the value of ϵ is based on the balance principle, whereby forces F and G are equal:

$$m_i\epsilon = m_i\gamma \cdot \Delta t^2 \tag{3.52}$$

where ϵ - spurious penetration depth, γ - gravitational acceleration ($9.82\frac{m}{s^2}$), Δt - update timestep.

The masses are equal and can be cancelled out. The penetration depth can then be calculated using:

$$\epsilon = \gamma \cdot \Delta t^2 \tag{3.53}$$

where ϵ - spurious penetration depth.

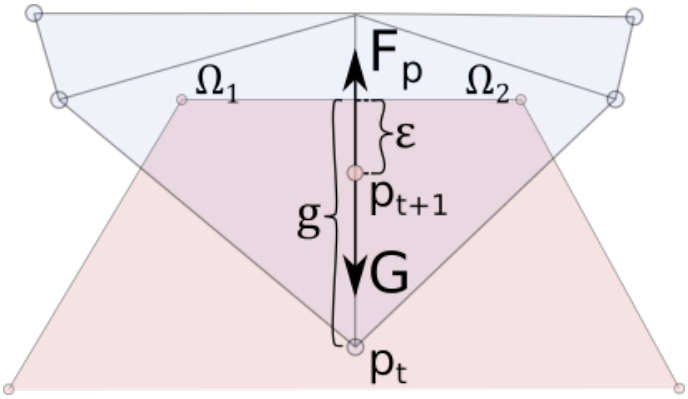


Figure 3.40: Two surfaces in contact with penalty based normal contact enforcement showing spurious penetrations of a node at position p_t. The position p_{t+1} located within the master surface is the position of the node after resolution. Gap g is before resolution and *epsilon* is the erroneous penetration after resolution.

These errors are negligible when the time step Δt is small and decreases quadratically with decreasing Δt. However, when the time steps are larger the spurious penetrations can lead to noticeable errors.

From Eq. 3.51 it can be seen that the force increases quadratically with a decreasing time step. The necessarily small time steps required to avoid the spurious penetration depth errors produce large forces which can have negative impact on the numerical stability of the contact resolution method.

Projection method for soft-vs-rigid body contact

The previously discussed research by Yastrebov (2013) and Cirak and West (2005) are examples of projection based contact methods. The reviews of these methods were expressed in Section 2.4.2. The main principle that distinguish the projection based methods is the use Dirichlet BC's to enforce non-penetration conditions.

The list of penetrating nodes is identified as the result of the contact detection stage. The nodes that are found to be inside the master surface are violating the non-penetration contact condition. In order to restore non-penetration it is necessary to move each of the penetrating nodes to the point of where the node would be positioned if the penetration was not allowed. The node projection stage

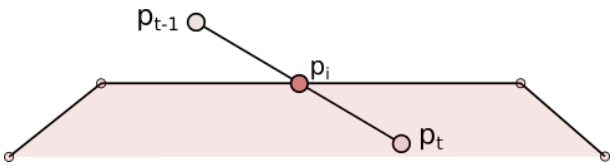


Figure 3.41: Shows a penetrating node and initial penetration point between frames

is dedicated for the purpose of finding the initial penetration point on the master surface and enforcing the non-penetration condition by moving the violating node out of the master surface.

For a given penetrating node the initial penetration point will be at a point between the current position at time t and the position of the node in the previous update frame at time $t - h$, where δt is time step. The exact time of penetration t_p could be used to find the initial penetration point by simply interpolating between the times t and $t - 1$. This interpolation process is demonstrated in Figure 3.41, where point p_i shows the initial penetration point using interpolation between points p_t and p_{t-1}. This method of finding the initial penetration point gives inaccurate results, except for the case of adhesive contact, whereby all the tangential movement of the penetrating node is close to zero.

A similar type of problem is studied in the area of continuous collision detection (CCD) (Redon et al., 2002). The distinguishing feature of CCD is the ability to find the point of initial penetration even with large time steps. Additionally, the method is required to find the position of the body after the penetration was removed, thus taking into account the movement of the body in the tangential plane. The CCD methods utilize swept versions of objections tested for collisions. For instance, a swept version of a point is a segment connecting the point's position between the updates. For a sphere, the swept version is a capsule. For the simpler case of a single point or a small sphere penetrating a plane the task is limited to finding the penetration point and the tangential slide using analytical formulae.

Figure 3.42 shows the case of a point, and equivalently of a small sphere, penetrating a plane. The point of initial penetration can be found using a simple segment and plane intersection. The final position of the point can then be found

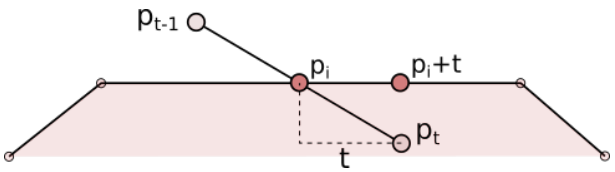


Figure 3.42: Position $p_i + t$ represents the position of the node when all penetration is removed and the tangential slide t is preserved.

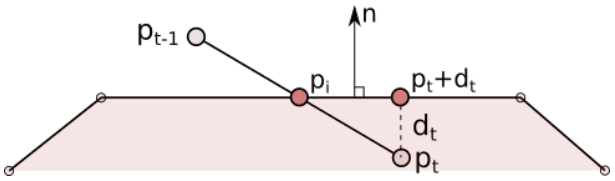


Figure 3.43: The same result as in Figure 3.42 is achieved when the node p_t is projected back onto the surface by adding $d_t n$

by adding the full tangential component of the displacement $p_t - p_{t-1}$ to the point p_i.

Note that the same final result can be obtained in a simpler manner by projecting the position of the point onto the plane as seen in Fig. 3.43. This approach results in considerably less computational effort with no loss of accuracy. The resulting final position $p_t + d_t$ is then found taking into account the distance from point p_t to the master surface plane d_t. Distance d_t is found according to Eq. 3.54 and the projected position according to Eq. 3.55.

$$d_t = (p_t - o) \cdot \hat{n} \tag{3.54}$$

where d_t - distance to plane of point p_t at time t, p_t - current nodal position, $\hat{n}$ - master plane normal, o - a point on the master plane (plane origin).

$$p_r = p_t - \hat{n} d_t \tag{3.55}$$

where p_r - final nodal position on master surface, p_t - current nodal position, $\hat{n}$ - master plane normal, d_t - distance to master plane.

3.3.6 Contact condition enforcement

Node projection in arbitrary meshes

The main task of the proposed contact method is the enforcement of the Signorini non-penetration condition for complex arbitrary meshes. For this purpose we utilize Dirichlet-like projection of penetrating slave nodes onto the master surface.

These contact conditions are specified per node and represent a planar approximation of the master surface from the node's perspective. Each penetrating node is assumed to be violating a single plane constraint. It is only allowed to be in one half-space or on the surface of the plane. Each of the planes are identified by the planes in which the corresponding master triangle lies. The master surface triangles are identified according to the ray-cast along the anti-normal direction as discussed in Section 3.3.4.

The contact conditions are formulated in the form of a list of penetrating nodes and corresponding contact plane. These contact conditions can be treated as part of the FE assembly in the form of boundary conditions. This implies that each contact condition is represented by a set of BC's that will after their enforcement satisfy the contact condition.

There are two different types of boundary conditions typically used in FEA: Dirichlet BC's and Neumann BC's. The two types in essence vary in the way they are enforced and the variables they are enforced upon. The nodes that are affected by a BC are called constrained nodes.

The Dirichlet boundary conditions, which are equivalent to essential boundary conditions in most cases, are imposed directly on the field variables. Such variables are represented by the nodal displacement, as an example, in case of the FEA used in childbirth simulation. This implies that the displacements for various constrained nodes are to be prescribed directly after each TLED update.

Neumann BC's, which are in most simple cases equivalent to the natural BC's, differ from Dirichlet BC's by prescribing the field variable values indirectly through the first or second derivatives. For instance, in case of childbirth FEA simulation the prescribed value of a Neumann BC will be the stress at the node. Note that the actual displacement (position) of the node can take any values and only the stress value at that position is enforced.

The method utilized for the purposes of our simulation is based on the Dirichlet BC enforcement. The method varies from the classical method by combining both the Dirichlet and Neumann BC's as a means to represent a contact condition. The method was initially described by Yastrebov (2013) and is called the partial Dirichlet-Neumann (pDN) contact method. Part of the BC is enforced along the normal of the contact surface as a Dirichlet BC, whereby the displacement is prescribed directly to ensure no penetration is present. The value for the BC, and subsequently the displacement of the node, is found using the node projection method described previously. All the tangential contributions of the contact condition, such as the friction between the surfaces in contact, are enforced as a Neumann in the form of frictional force (acceleration) acting along the tangential plane.

A projection boundary condition is assigned to a node to prevent it from entering a half-space of a plane. For non-frictional contact the normal of the plane $\hat{n}$ and the distance to origin d_o are sufficient to define a valid projection boundary condition.

The non-penetration into the negative half-space is enforced by calculating the distance d of the node to the plane p. The distance must be always positive. Otherwise the projection of the node on the plane p is found and the position of node is prescribed using Dirichlet-like enforcement.

Persistent contact conditions

o In the context of amortized contact detection it is possible that the node, moved out of penetration, can immediately re-penetrate the surface due to the deformation induced internal forces. During the TLED sub-steps in between the contact detection updates it is possible that the previously moved nodes will move into the master surface. This situation is shown in Figure 3.44. The initial penetration is detected and resolved by activating the plane contact constraint. The constraint pushes the penetrating nodes out of penetration. After the successful non-penetration condition enforcement, the plane constraint is disabled. At time $t + 2$ the node is free without any constraints and the internal forces from the neighbouring elements can push it back into penetration at time $t + 3$. The effect

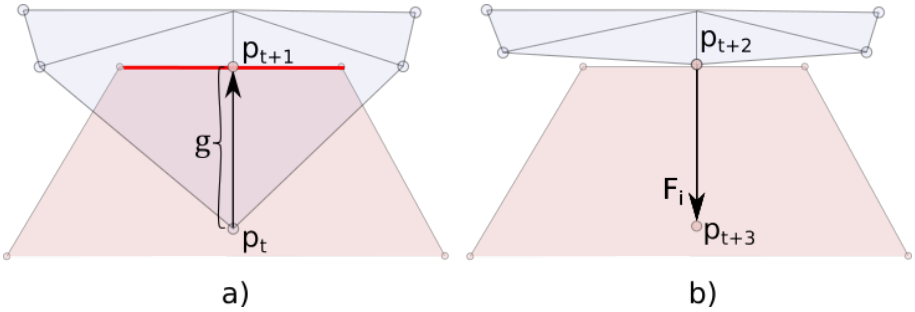


Figure 3.44: Left: penetrating node p_t with gap g is moved out of penetration to position p_{t+1}. Right: same node currently not penetrating the master is subjected to internal forces from adjacent finite elements pushing it back to its reference configuration. p_{t+3} shows the next node position, which brings the node back into the state penetration.

of this would be the node oscillating in and out of penetration, as the penetration from the internal force will be detected during the next contact detection update and resolved.

One approach to prevent such re-penetrations is to perform contact detection on the projected future nodal positions. Alternatively, the contact plane constraint can be persisted even after the penetration has been resolved for a duration of time. Note how in case of projection contact boundary conditions the node after resolution can be prevented from re-entering master surface by preserving the contact condition for several future updates. Figure 3.45 demonstrates how the plane constraint was left intact even after the node retreated from the master surface. The nature of projection contact boundary conditions is that they do not affect the node while its on the *outside* halfspace of the projection plane (that is outside the master surface) even while the condition is active. In Figure 3.45 (c) the node **p** has an active plane constraint attached and yet is able to move freely as long as it does not penetrate into the negative plane halfspace (the master volume).

The plane constraints are persisted for a limited number of frames. Each frame in which the constraint was active accounts for a single *hit*. For the simulations of childbirth reported in this thesis, the number of *hits* per plane constraint before it is deleted was set to 6.

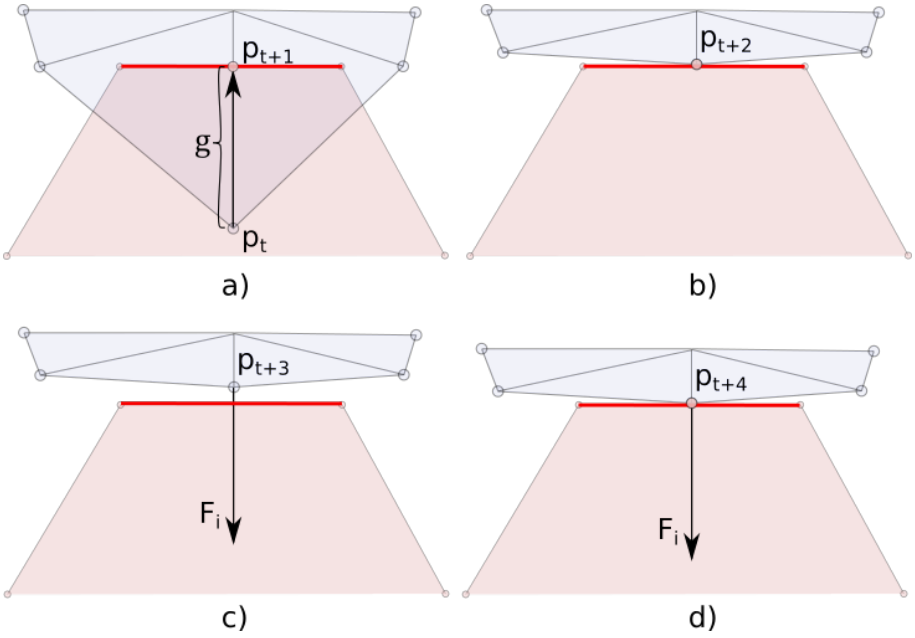


Figure 3.45

3.3.7 Contact force evaluation

Contact force in projection based methods

The previous section showed how the projection based contact is used to enforce non-penetration conditions. The penetrating nodes can be reliably identified now and projection based Dirichlet boundary conditions can be used to enforce the contact conditions. No contact forces need to be calculated or considered altogether in the simple cases when the contact is evaluated between a static rigid object interacting with a deformable FE mesh.

However, there is an issue that occurs when the contact is being evaluated between a deformable FE mesh and a dynamic rigid body. In this case the non-penetration conditions on the deformable slave FE mesh are imposed in exactly the same way as described, but the dynamic rigid master body also needs to take part in the interaction. The master body needs to receive a contact force appropriate to the type of contact being evaluated. The projection based contact requires evaluating no such force and therefore the contact force that should act on the master is unknown.

The traction force representing the surface traction between the two interacting bodies (slave and master) can evaluated by integrating the stresses on the FE mesh surface over the contact area. The surface stresses need to be calculated based on the stresses within the volumetric finite elements. The internal forces acting on the slave nodes from the surrounding nodes actually contain the information about the surface stresses need to evaluate as will be shown further. Figure 3.46 demonstrates the gap and the projection based contact approaches for calculating contact reaction forces. The penalty based approach calculates reaction forces based on nodal penetrations, emulating linear springs in the contact interface, as seen in Figure 3.46 (a). Both the penetrating nodes and the master surface receive the contact reaction forces. In case of the projection based approach in Figure 3.46 (b), the reaction forces are never calculated for the penetrating nodes as they are projected onto the master surface. The stresses in the outermost slave elements in the contact interface present a convenient way to evaluate the contact stresses, which can in turn be integrated to obtain the contact force acting on the master surface.

The principle of using the stresses within the outermost slave object elements as the source of the contact reaction forces is explored further in this section. The Finite Volume Method (FVM) is a method similar to FEM. Where the FEM features stresses occurring within finite elements, the FVM approach is to look at the stresses on the surfaces of the elements. It can be seen how this approach can be relevant to the case of the presented projection method where the contact forces are calculated based on the surface stresses.

Traction force

The proposed method relies on using the reaction force in the surface nodes as the contact force. The reaction force in surface elements can be calculated by integrating the stress occurring in the contact surface due to deformation imposed on it during the projection based non-penetration enforcement:

$$f_c = \int_A \sigma \boldsymbol{n} dA \tag{3.56}$$

where f_c - contact reaction force; A - contact surface; $\boldsymbol{n}$ - contact surface normal.

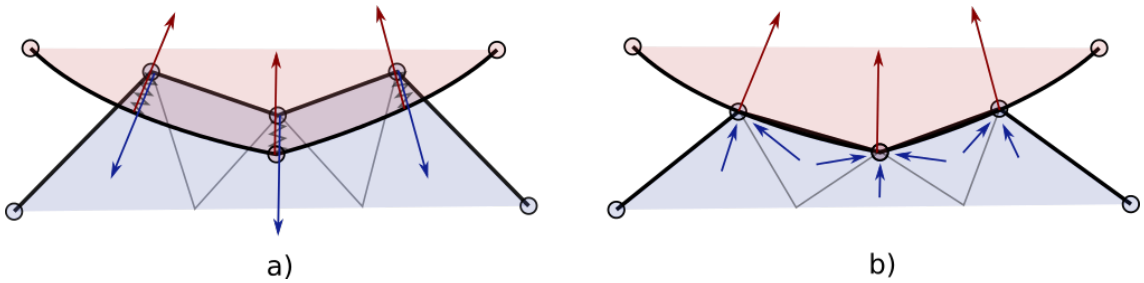


Figure 3.46: A penalty based contact resolution approach, where the reaction forces are evaluated based on nodal penetrations (a). The contact force is exerted on both the slave and the master in order to resolve contact (blue and red arrows). A projection based contact does not rely on contact forces to resolve contact (b). Penetrations are removed by projecting nodes onto the master surface. The contact force acting on the master is then calculated based on the stresses in the outermost elements in the contact region (blue arrows).

Note that for a linear element the normal remains constant, the contact surface is fixed and the stress is constant over the element. Thus the integral of Eq. 3.56 evaluates to:

$$f_c = \sigma \boldsymbol{n} A, \tag{3.57}$$

Traditionally, FEM utilizes integration of stress over element volume in order to evaluate the nodal forces. Thus the nodal forces are expressed in terms of quantities varying over the element volume. Our contact method, however, relies on evaluating nodal forces in terms of quantities varying over the element surface. In order to formulate the desired relationship it is required to correlate the quantities in the volume to the surface.

Divergence theorem

The divergence theorem or Gauss' theorem presents a way of correlating the behaviour of a vector field on the surface of a domain to its behaviour throughout the domain volume. This relationship is shown in Eq. 3.58:

$$\int_V (\nabla \cdot \boldsymbol{F}) dV = \oint_A (\boldsymbol{F} \cdot \boldsymbol{n}) dA, \tag{3.58}$$

where V - domain volume; A - domain boundary surface[1]; $\boldsymbol{F}$ - continuously differentiable vector field; $\boldsymbol{n}$ - vector field of outward pointing unit normals of the domain boundary.

The left-hand side is a volume integral, sometimes written as a triple integral, and the right-hand side is a surface contour integral. Note the circle in the integral, indicating that the surface needs to be closed. The Finite Volume Method (FVM) is a method that can be used to connect the surface based nodal force formula with a volume based one using the divergence theorem.

Finite Volume Method - FVM

The FVM provides a geometrically intuitive way of evaluating the stress based contact force (Teran et al., 2003). For simplicity we will initially provide the FVM formulation for a 2D triangle and then extend it to a 3D tetrahedron.

Consider a mesh consisting of triangular elements shown in Fig. 3.47. The boundary of the domain Ω is represented by $\partial\Omega$. In case of a linear constant-strain element the Cauchy stress σ is constant over the element. This results in the divergence of σ being zero:

$$\nabla \cdot \sigma = 0 \tag{3.59}$$

The left-hand side volume integral from the divergence theorem in Eq. 3.58 is then zero. This results in the sum of all domain border integrals also being zero:

$$\int_{\partial\Omega_1} \sigma \boldsymbol{n} dA + \int_{\partial\Omega_2} \sigma \boldsymbol{n} dA + \int_{\partial T_1} \sigma \boldsymbol{n} dA + \int_{\partial T_2} \sigma \boldsymbol{n} dA = 0 \tag{3.60}$$

Eq. 3.60 can be transformed to equate the internal integrals to the negative of the external integrals:

$$\int_{\partial\Omega_1} \sigma \boldsymbol{n} dA + \int_{\partial\Omega_2} \sigma \boldsymbol{n} dA = -(\int_{\partial T_1} \sigma \boldsymbol{n} dA + \int_{\partial T_2} \sigma \boldsymbol{n} dA) \tag{3.61}$$

The integral over the whole of the neighbourhood domain boundary $\partial\Omega$ is equal to the sum of integrals over all incident edges to the node. The sum of integrals over internal borders $\partial\Omega_i$ can be replaced by the sum of integrals over

[1]Note that the domain is in general denoted by Ω and the domain boundary by $\partial\Omega$.

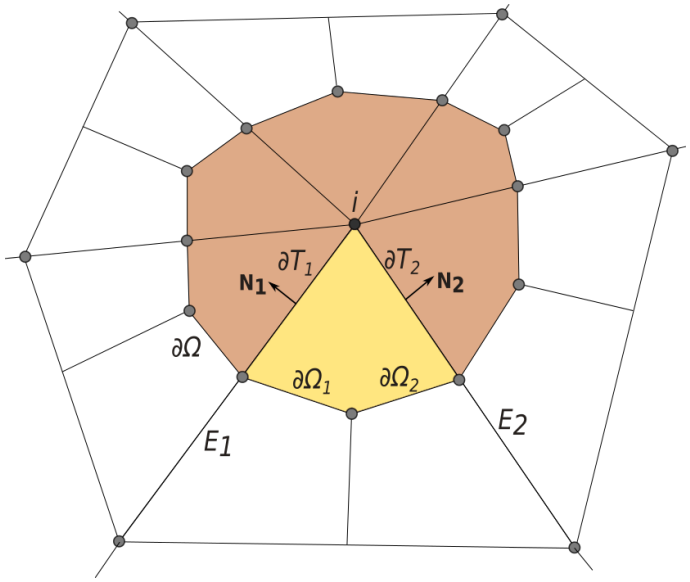


Figure 3.47: Finite volume method integration regions showing the nodal neighbourhoods (highlighted) in which stress is integrated.

external edges ∂T_i. Thus, the nodal internal force f_i is then calculated according to Eq. 3.62:

$$f_i = \oint_{\partial\Omega} \sigma \boldsymbol{n} dA = -\sum_j \int_{\partial T_j} \sigma \boldsymbol{n} dA, \tag{3.62}$$

where j enumerates all incident edges at node i.

Recall that the internal segments $\partial\Omega_1$ and $\partial\Omega_2$ were chosen to connect the edge midpoints with the center. This implies that the lengths of ∂T_1 and ∂T_2 are equal to half the lengths of the corresponding edges, denoted as E_1 and E_2. Evaluating the external integrals we get Eq. 3.63:

$$f_i = -\sum_{j=1}^{n} \frac{1}{2} \sigma (E_j \boldsymbol{n}_j + E_{(j+1) \bmod n} \boldsymbol{n}_{(j+1) \bmod n}) \tag{3.63}$$

where f_i - nodal force at node i; j enumerates all incident elements at node i; E_j and $E_{(j+1) \bmod n}$ - edges of element j converging at node i; n - the number of edges/elements of the surrounding polygon of node i; $\boldsymbol{n}_j$ and $\boldsymbol{n}_{(j+1) \bmod n}$ - normals to the edges of element j converging at node i.

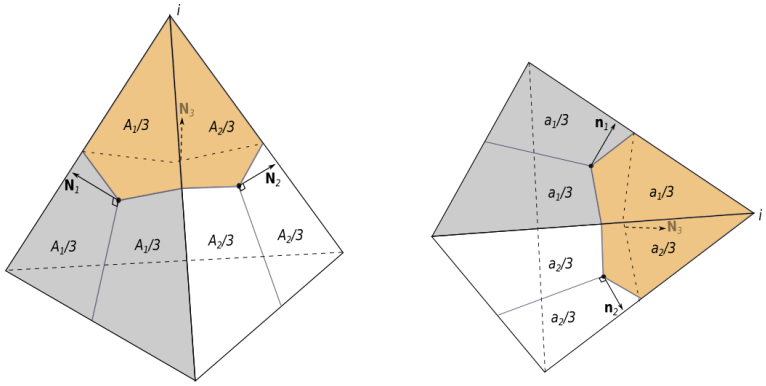


Figure 3.48: A tetrahedron in the initial underformed configuration (left) and deformed configuration (right). FVM integration over a 3D tetrahedron domain around i-th node (highlighted). A_j and a_j are the areas of the j-th face in the reference and deformed configuration respectively. $\boldsymbol{N}_j$ and $\boldsymbol{n}_j$ are the normals of the j-th face in the reference and deformed configuration respectively. Note that the back and bottom faces with index 3 and 4 are present but not marked.

A similar procedure can be applied to the case of a three dimensional tetrahedral element mesh. Consider the tetrahedron shown in Fig. 3.48. It can be seen that each tetrahedral node has three faces converging at it. Therefore, in Eq. 3.62 the right hand side will represent a sum of integrals over each of the portion of the incident faces of the node. Note that to calculate the full nodal force, j is varying over all incident faces from all neighbouring elements and not only the faces of a single element in Fig. 3.48.

As seen in Fig. 3.48, the integration region boundaries connect the midpoints of the edges with the face midpoints. This results in the areas of each of the integration points on a given face being equal to one third of the total face area. This allows us to obtain the equation for the tetrahedron in a similar way to Eq. 3.63 for the triangle. Integrating constraint stress over a constant area and normal gives us Eq. 3.64 for the tetrahedron:

$$f_i = -\sum_{j=1}^{n} \frac{1}{3}\sigma_j(a_{j,1}\boldsymbol{n}_{j,1} + a_{j,2}\boldsymbol{n}_{j,2} + a_{j,3}\boldsymbol{n}_{j,3}) \tag{3.64}$$

where f_i - nodal force at node i; j enumerates all incident elements at node i; σ_j - Cauchy stress in element j; $a_{j,1}$, $a_{j,2}$ and $a_{j,3}$ - areas of faces of element j

converging at node i; n - the number of elements of the surrounding volume of node i; $\boldsymbol{n}_{j,1}$, $\boldsymbol{n}_{j,2}$ and $\boldsymbol{n}_{j,3}$ - normals to the faces of element j converging at node i.

Following the Lagrangian formulation, we express the nodal force values in terms of stresses parametrized in material coordinates. For this purpose we express the Cauchy stress σ in terms of the first Piola-Kirchhoff stress $\boldsymbol{P}$ (Holzapfel, 2000):

$$\sigma = J^{-1}\boldsymbol{P}\boldsymbol{F}^T \tag{3.65}$$

where σ - Cauchy stress, J - determinant of $\boldsymbol{F}$

After substituting σ in 3.64 we get:

$$f_i = -\sum_{j=1}^{n} \frac{1}{3}\boldsymbol{P}_j J_j^{-1}\boldsymbol{F}_j^T(a_{j,1}\boldsymbol{n}_{j,1} + a_{j,2}\boldsymbol{n}_{j,2} + a_{j,3}\boldsymbol{n}_{j,3}) \tag{3.66}$$

The areas and normals in the current (spatial) configuration can be related to those in the reference (material) configuration using Nanson's formula (Holzapfel, 2000):

$$A\boldsymbol{N} = J^{-1}\boldsymbol{F}^T a\boldsymbol{n} \tag{3.67}$$

This gives us the following equation for the nodal force in terms of the first Piola-Kirchhoff stress and areas and normals in the reference configuration as seen in Fig. 3.48:

$$f_i = -\sum_{j=1}^{n} \frac{1}{3}\boldsymbol{P}_j(A_{j,1}\boldsymbol{N}_{j,1} + A_{j,2}\boldsymbol{N}_{j,2} + A_{j,3}\boldsymbol{N}_{j,3}) \tag{3.68}$$

where f_i - force on node i; $A_{j,l}$ - area of face l from element j; $\boldsymbol{N}_{j,l}$ - normal of face l from element j; n - the number of elements of the surrounding volume of node i.

Each expression inside the summation in Eq. 3.68 represents the individual incident element contribution to the nodal force f_i. The quantities for the incident faces $A_{j,l}$ and $\boldsymbol{N}_{j,l}$ are labelled from the perspective of the node i, with j varying over all the incident elements. The label l is used for each of the 3 faces of element j that are converging at node i. When the faces contributing to the internal force

of node i are considered from the perspective of the element j then they are labelled as $A_k \boldsymbol{N}_k$.

The sum of area and normal products in column vectors $b_j = \frac{1}{3}\sum_{k \neq j} A_k \boldsymbol{N}_k$ can be precomputed and collected in a matrix of column vectors:

$$\boldsymbol{B}_m = -\frac{1}{3}\begin{bmatrix} A_2\boldsymbol{N}_2 + A_3\boldsymbol{N}_3 + A_4\boldsymbol{N}_4 \\ A_1\boldsymbol{N}_1 + A_3\boldsymbol{N}_3 + A_4\boldsymbol{N}_4 \\ A_1\boldsymbol{N}_1 + A_2\boldsymbol{N}_2 + A_4\boldsymbol{N}_4 \\ A_1\boldsymbol{N}_1 + A_2\boldsymbol{N}_2 + A_3\boldsymbol{N}_3 \end{bmatrix}^T = \begin{bmatrix} b_1^T \\ b_2^T \\ b_3^T \\ b_4^T \end{bmatrix} \tag{3.69}$$

According to the divergence theorem in Eq. 3.58 we can relate the area weighted normals of any face on the tetrahedron to the sum of all the other area weighted face normals. Physical interpretation of this is that a tetrahedron filled with pressurized gas will experience zero net force from pressure on all its internal faces. This results in the following relationship:

$$A_1\boldsymbol{N}_1 + A_2\boldsymbol{N}_2 + A_3\boldsymbol{N}_3 + A_4\boldsymbol{N}_4 = 0 \tag{3.70}$$

Eq. 3.70 can be used to simplify 3.69 to be:

$$\boldsymbol{B}_m = \frac{1}{3}\begin{bmatrix} A_1\boldsymbol{N}_1 \\ A_2\boldsymbol{N}_2 \\ A_3\boldsymbol{N}_3 \\ A_4\boldsymbol{N}_4 \end{bmatrix}^T = \begin{bmatrix} b_1^T \\ b_2^T \\ b_3^T \\ b_4^T \end{bmatrix} \tag{3.71}$$

All the force contributions to the nodes within the element can now be collected in a matrix:

$$f^e_{FVM} = \begin{bmatrix} g_1 & g_2 & g_3 & g_4 \end{bmatrix}^T = \boldsymbol{P}\boldsymbol{B}_m = \boldsymbol{P}\begin{bmatrix} b_1 & b_2 & b_3 & b_4 \end{bmatrix} \tag{3.72}$$

where f^e_{FVM} - element nodal force contribution matrix, g_i - nodal force contribution to node i within element, $\boldsymbol{B}_m$ - matrix of incident face area and normal products in material coordinates.

Nodal contact force calculation

Previously 3.2 we have shown how the individual element contribution to the nodal forces can be calculated (See Eq. 3.28) using the TLED formulation. Note, that this calculation is performed regardless of contact conditions for all nodes of the FE mesh for the purposes of the deformable body simulation. We now propose to use the results of this calculation for the reaction force calculation and avoiding the extra calculation using Eq. 3.64. For this purpose it is necessary to demonstrate complete equivalence of the two formulations.

It can be shown that the FVM formulation is exactly equivalent to the FEM TLED equation for calculating the nodal forces in case of a linear constant-strain tetrahedron. For this purpose we can initially assume that the resultant matrix of nodal forces from both formulations are equal. Note that the TLED formulation results in column-wise orientated force vectors in f^e, therefore a transposition is required resulting in Eq. 3.73.

$$f^e_{FVM} = (f^e)^T \tag{3.73}$$

where f^e_{FVM} - element nodal force contribution matrix from FVM formulation; f^e - element nodal force contribution matrix from TLED formulation.

Equating the actual expressions for nodal contributions gives Eq. 3.74:

$$\boldsymbol{PB}_m = (V_0 \partial h \boldsymbol{S} \boldsymbol{F}^T)^T \tag{3.74}$$

where ∂h - shape function derivative matrix with respect to material coordinates, $\boldsymbol{F}$ - deformation gradient.

After transposing, Eq. 3.74 becomes:

$$\boldsymbol{PB}_m = V_0 \boldsymbol{F} \boldsymbol{S}^T \partial h^T \tag{3.75}$$

The second Piola-Kirchhoff stress is symmetric, $\boldsymbol{S} = \boldsymbol{S}^T$, and is related to the first Piola-Kirchhoff stress through the deformation gradient, $\boldsymbol{P} = \boldsymbol{F}\boldsymbol{S}$. Substituting into 3.75 gives:

$$\boldsymbol{PB}_m = V_0 \boldsymbol{P} \partial h^T \tag{3.76}$$

and

$$\boldsymbol{B}_m = V_0 \partial h^T \tag{3.77}$$

Equation 3.77 defines the relationship between the two methods. In order to prove the equivalence of the two methods it is required to demonstrate that the right-hand side of the equation evaluates to the right-hand side of Eq. 3.69.

In order to demonstrate the equivalence it is necessary to investigate the contents of the shape function derivatives $\partial \boldsymbol{h}$. Recall Eq. 3.31, repeated and relabelled here as Eq. 3.78, for convenience:

$$\frac{\partial \xi}{\partial \boldsymbol{X}} = \boldsymbol{J}^{-1} = \frac{1}{6V} \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \tag{3.78}$$

The triples of values can be combined into vectors $s_i = \begin{bmatrix} a_i & b_i & c_i \end{bmatrix}$.

- The length of the vector $|s_i| = \sqrt{a_i^2 + b_i^2 + c_i^2} = 2A_i$, where A_i - area of the face opposite to node i.
- The normalized vector $\hat{s}_i = s_i / |s_i|$ represents the unit vector of the opposing face.

Additionally, it can be shown that an inverse of a matrix consisting of three column vectors can be represented as a matrix of cross-products scaled by the inverse of the vector triple product:

$$\boldsymbol{C}^{-1} = \begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix}^{-1} = \frac{1}{c_1 \cdot (c_2 \times c_3)} \begin{bmatrix} c_2 \times c_3 & c_3 \times c_1 & c_1 \times c_2 \end{bmatrix}^T \tag{3.79}$$

where $\boldsymbol{C}$ - matrix of column vectors c_i.

The triple product is the matrix determinant giving six times the tetrahedral volume. Therefore, s_i is the twice-area weighted internal unit normal of the face opposite to node i, which is the same as the cross product of two of the same

face's edges. Knowing that $s_i = 2A_i\boldsymbol{N}_i$ we get:

$$\frac{\partial \xi}{\partial \boldsymbol{X}} = \frac{1}{3V}\begin{bmatrix} A_1\boldsymbol{N}_1 \\ A_2\boldsymbol{N}_2 \\ A_3\boldsymbol{N}_3 \end{bmatrix} \tag{3.80}$$

Substituting Eq. 3.78 and Eq. 3.80 into Eq. 3.36 and evaluating the matrix multiplication gives:

$$\partial \boldsymbol{h} = \frac{1}{3V}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -1 & -1 \end{bmatrix}\begin{bmatrix} A_1\boldsymbol{N}_1 \\ A_2\boldsymbol{N}_2 \\ A_3\boldsymbol{N}_3 \end{bmatrix} = \frac{1}{3V}\begin{bmatrix} A_1\boldsymbol{N}_1 \\ A_2\boldsymbol{N}_2 \\ A_3\boldsymbol{N}_3 \\ -(A_1\boldsymbol{N}_1 + A_2\boldsymbol{N}_2 + A_3\boldsymbol{N}_3) \end{bmatrix} \tag{3.81}$$

Using Eq. 3.70 gives the shape function derivatives for an iso-parametric linear tetrahedron in terms of face areas and normals:

$$\partial \boldsymbol{h} = \frac{1}{3V}\begin{bmatrix} A_1\boldsymbol{N}_1 \\ A_2\boldsymbol{N}_2 \\ A_3\boldsymbol{N}_3 \\ A_4\boldsymbol{N}_4 \end{bmatrix} \tag{3.82}$$

Eq. 3.77, after substitution of Eq. 3.82, then gives the same result as Eq. 3.71, demonstrating the equivalence of the two methods:

$$\boldsymbol{B}_m = V\partial h^T = \frac{1}{3}\begin{bmatrix} A_1\boldsymbol{N}_1 \\ A_2\boldsymbol{N}_2 \\ A_3\boldsymbol{N}_3 \\ A_4\boldsymbol{N}_4 \end{bmatrix}^T \tag{3.83}$$

Reaction forces Nodal force contributions of an element e are evaluated using:

$$f^e = \boldsymbol{P}\boldsymbol{B}_m = \frac{1}{3}\boldsymbol{P}\begin{bmatrix} A_1\boldsymbol{N}_1 \\ A_2\boldsymbol{N}_2 \\ A_3\boldsymbol{N}_3 \\ A_4\boldsymbol{N}_4 \end{bmatrix}^T \tag{3.84}$$

The first Piola-Kirchhoff stress relates the normal in the reference configuration to the traction force in the current configuration (Holzapfel, 2000):

$$\boldsymbol{T} = \boldsymbol{P}\boldsymbol{N} \tag{3.85}$$

where $\boldsymbol{T}$ - traction force in current configuration; $\boldsymbol{N}$ - area normal in the reference configuration.

The force acting on a given area A is calculated according to:

$$f = \boldsymbol{T}A = \boldsymbol{P}\boldsymbol{N}A \tag{3.86}$$

where A - the area in the reference configuration.

By substituting Eq. 3.86 into Eq. 3.84 we get:

$$f^e = \frac{1}{3}\begin{bmatrix} \boldsymbol{P}A_1\boldsymbol{N}_1 & \boldsymbol{P}A_2\boldsymbol{N}_2 & \boldsymbol{P}A_3\boldsymbol{N}_3 & \boldsymbol{P}A_4\boldsymbol{N}_4 \end{bmatrix} = \begin{bmatrix} g_1 & g_2 & g_3 & g_4 \end{bmatrix}, \tag{3.87}$$

where g_i - column vectors of element nodal force contribution to node i; $A_i\boldsymbol{N}_i$ - are column vectors allowing for compatibility with $\boldsymbol{P}$.

According to Eq. 3.87 the contact reaction force on a tetrahedral face from Fig. 3.49 is given by element's nodal force contribution to the opposite node. Note that using Eq. 3.70 this is equivalent to the inverse of the nodal force contributions to all the other nodes, which actually represent the face in contact:

$$R_i = f_i = \frac{1}{3}\boldsymbol{P}A_i\boldsymbol{N}_i = -\frac{1}{3}\sum_j \boldsymbol{P}A_j\boldsymbol{N}_j \tag{3.88}$$

where R_i - contact reaction force on tetrahedral face i; f_i - element nodal force contribution to node i of the element; j - varies from 1 to 4 and is not equal i.

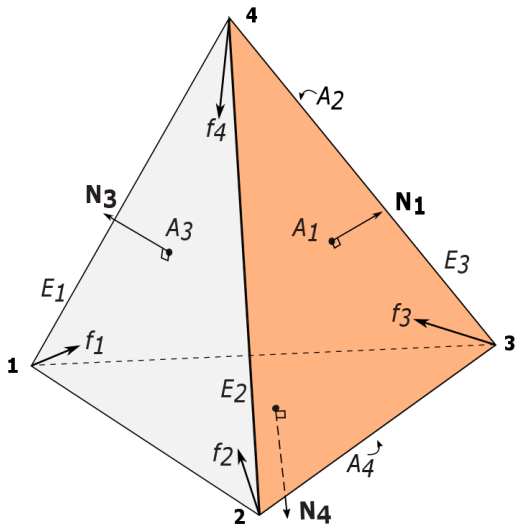


Figure 3.49: A tetrahedron in contact on face 1 (highlighted). E_i are edges and A_i and $\boldsymbol{N}_i$ are face areas and normals, respectively. f_i are nodal force contributions from the element

We have shown using the above derivations for a tetrahedral face in contact, that it is justified to use the inverse of the sum of forces from nodes connected to the face as the contact reaction force. In a friction-less setting, the reaction force is projected onto the normal, resulting in contact forces acting only along the contact normal. Based on this we can utilize the TLED internal reaction force values from the latest TLED update with no extra computational cost for contact force calculation apart from the projection. As seen in Figure 4.3, the TLED update stage is performed independent of contact. During the TLED update the internal forces for all the nodes in the FE mesh will be calculated, including the ones currently in contact. These nodal force values can be extracted and used as the contact force values, which leads to an improvement in terms of performance.

Chapter 4

Implementation and Software Framework

4.1 Software Framework

4.1.1 General Concepts

The BirthView simulation system is written in the C++ language. The majority of code is written in a cross-platform manner, which allows compiling the software for most of the popular platforms:

1. Microsoft Windows [1]
2. Apple OSx [2]
3. Linux [3]
4. Other platforms can be supported subject to minor adaptations

All general purpose GPU calculations are implemented using the OpenCL (Stone et al., 2010a) framework, which also ensures better portability in terms of operating system and the GPU vendor. More information about the details of the OpenCL implementation are provided in Sec. 4.2

[1] https://www.microsoft.com/en-gb/windows
[2] http://www.apple.com/macos/
[3] https://www.linuxfoundation.org/

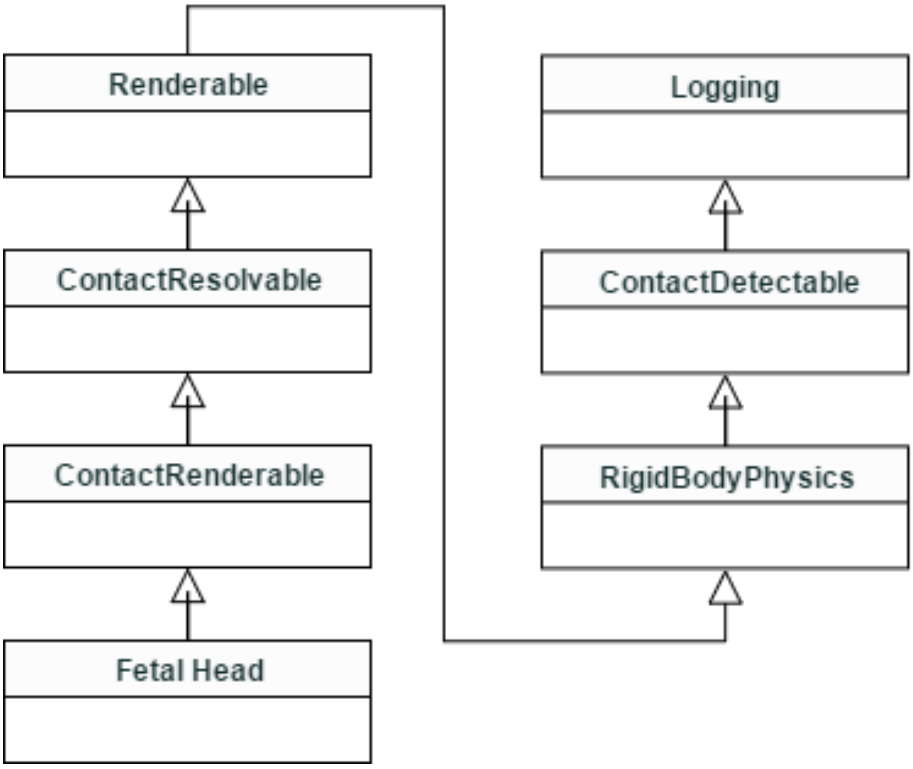


Figure 4.1: The fetal head entity described using an Object-Oriented Programming (OOP) hierarchy. Multiple layers of inheritance are required.

4.1.2 Entity Component System

The core architecture of the BirthView simulation system is built according to the Entity Component System (ECS) architectural pattern. As the name suggests, the two main concepts of the pattern are *entity*, *component* and a *system*. A good description of this principle is provided by Gaul (2013) and Gregory (2009).

An *entity* identifies one of the discernible objects in the simulation, such as pelvis, skull, pelvic floor, etc. Note, however, that an *entity* by itself does not represent these objects, but rather gives them an identity. Additionally, it does not need to identify a visible object as the ones listed above, but it may also be the camera, an arbitrary attachment point, the origin of a force field, etc. An *entity* will typically encapsulate its position, orientation, scaling and name.

The *components* are the basic blocks that add features and behaviour to an

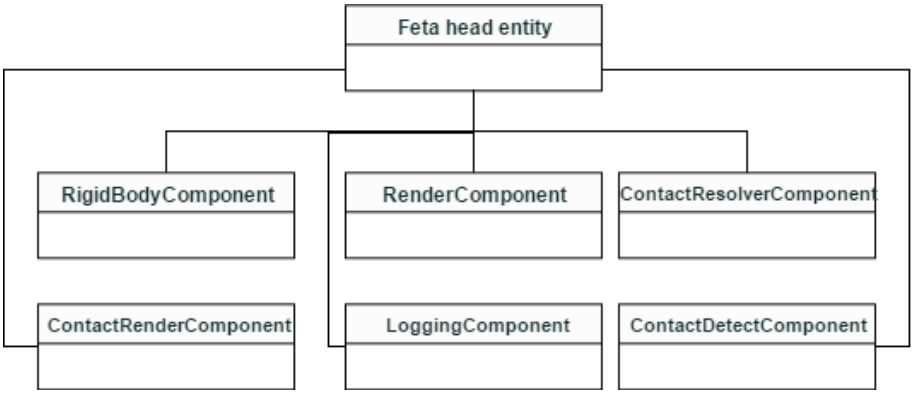


Figure 4.2: Entity component system (ECS) diagram of the fetal head. Note that no inheritance is required and the lines only indicate references.

entity. As mentioned, an *entity* without attached *components* is merely a dummy object without any functionality, but adding components will add additional behaviour to it. Adding a specific set of components to an entity can be used to model objects of any type.

The traditional object oriented programming (OOP) approach using hierarchies of classes will quickly explode in terms of complexity. Figure 4.1 shows how the fetal head entity can be described using an OOP hierarchy. Note that a deep hierarchy is required to describe all the functionalities of the fetal head. A lot of redundancy is introduced and changing the functionality by modifying the hierarchy is complicated. Consider now that another object, for instance the pelvis, needs to be modelled. The pelvis needs to have rendering functionality, but does not need to act like a dynamic rigid body. Inheriting from the `Renderable`, will implicitly inherit from the other classes in the hierarchy. This is a redundant and fragile design.

The main principle of ECS is that composition should be preferred over inheritance. Figure 4.2 shows the same functionality of the fetal head implemented using the ECS approach. No complex hierarchy is required and modifying the functionality of the entity is done simply by adding or removing components.

A *system* is meant to encapsulate a distinct functionality of the system as whole. Examples of systems implemented in our simulation system:

1. *Rendering system* - responsible for all rendering in the system; all draw calls, render state changes and texture manipulators are processed here
2. *Physics system* - responsible for all basic rigid physics updates
3. *GUI system* - responsible for all GUIs
4. *Manipulator system* - responsible for all the manual translation, rotation and scaling using manipulators and their rendering
5. *Selection system* - responsible for mouse picking and keeping track of selected entities
6. *Camera system* - responsible for storing and updating all cameras present in a simulation
7. *Mouse system* - responsible for mouse state storage and updates
8. *Keyboard system* - responsible for keyboard state storage and updates
9. *Windowing system* - responsible for creating windows, dialogs, resize events, and most of user input; contains most of the platform specific code (Windows, Linux, OSX)
10. *Collision system* - responsible for all collision detection in the system

The functionality encapsulated by a system is not specific to a given entity, but rather to all entities that match certain criteria. Additionally, functionality of a system may not be directly related to any *entity* at all, such as the functionality added by the `Keyboard` or `Mouse` systems. For instance, only entities with an attached `Collider` *component* will be used by the *Collision system* to perform collision detection amongst them.

4.1.3 System Concurrency

For the purposes of better performance some of the simulation functions need to be run in parallel. In particular most of the systems described in the previous section can be run in parallel as they have no dependency on each other.

The `UpdateSystem` is present in BirthView as the main source of concurrent updates. The *observer pattern* from Gamma et al. (1994) is used here, where the subject is the update event and observers are all systems that require to perform some actions concurrently to each other.

There are, however, cases when some systems require access to the results of another system's update, thus creating a dependency of one system update on another. An example of such a dependency is any system that needs to render something, needs to enqueue all the render calls in a synchronous manner.

One solution to this problem is using *mutexes* to block during writes, whereby any system will lock a *mutex* and release it when concurrent access is allowed. The disadvantage to this approach is that in a complex scenario the multitude of *mutexes* can generate *live-locks* or *dead-locks*, thus stalling the systems involved.

An alternative solution was implemented in BirthView. As most of the functionality of the systems does not have a dependency on other systems, it is possible to leave that independent part in the threads running in parallel. However, the part of the functionality that has inter-system access dependencies may not be run concurrently. Therefore, a *sync* event was added to the `UpdateSystem`, which calls the observer's callbacks in a sequential manner. Additionally, this event is only triggered after joining (synchronizing) all the parallel threads.

A basic representation of this approach is shown in Fig. 4.3. Note how the parallel stage lasts as long as the longest task (also known as the *span*), which is in this case the TLED update.

The current implementation of the described parallel execution is based around the `std::thread` class provided by the C++11 standard library (Meyers, 2014). The observer callbacks in the form of `std::function` class instances are in an `std::vector`. The point where the parallel execution is initiated is called a *fork*. The *fork* is implemented by spawning a set of `std::thread` instances each tasked to invoke the callbacks.

The most optimal split of a workload is when the task is divided to all CPU cores equally(Meyers, 2014). When the number of tasks is less than or equal to the number of CPU cores, each core is given one of the tasks, in which case not all cores will receive some load. When the number of threads is more than the number of cores available on the CPU, the CPU will be required to do time-

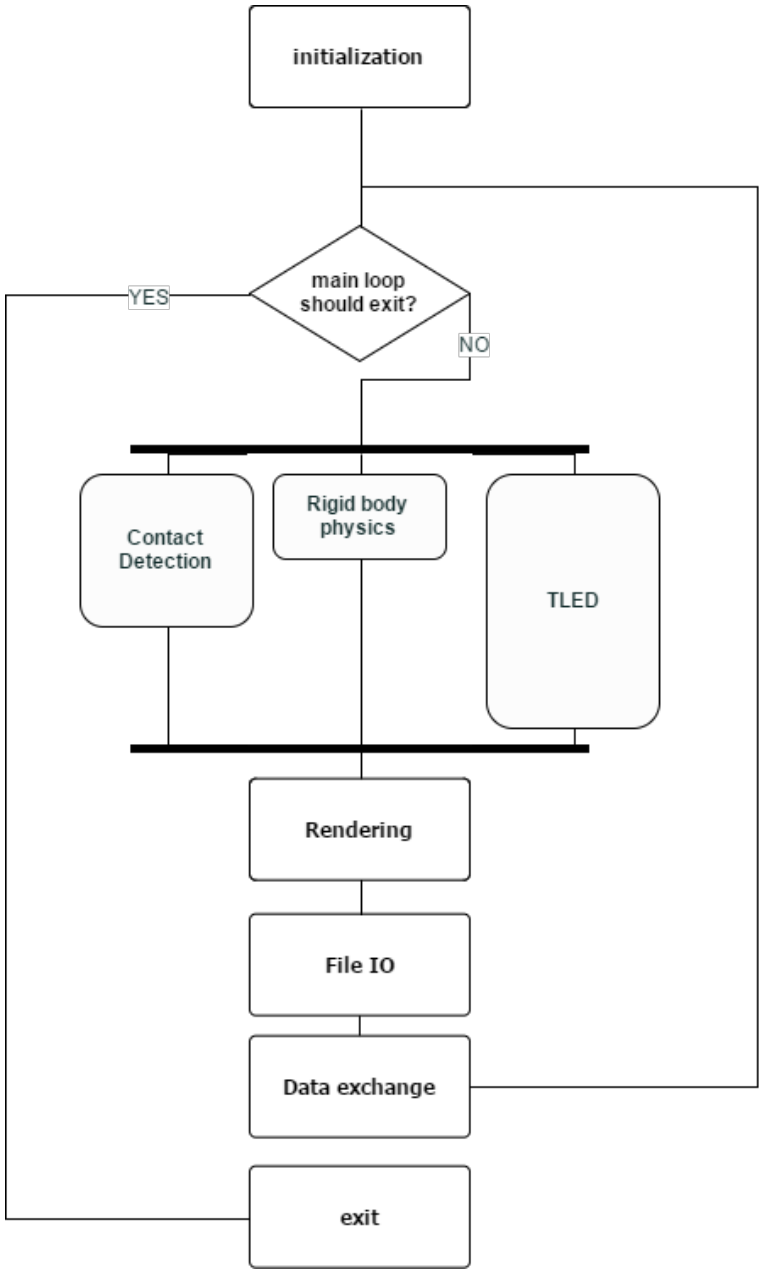


Figure 4.3: A set of concurrent updates and a sync event at the end of each frame showing all systems synchronizing at it. Further tasks are executed synchronously.

slicing and context switches. In this situation a lot of CPU time is spent without performing useful work, which is detrimental to the overall performance of the simulation system. In order to avoid this detrimental workload on the CPU, it is beneficial to only spawn as many threads as there are cores.

The workload can then be distributed between the threads by distributing the tasks equally, when the number of tasks is exactly divisible by the number of cores. In the opposite case the first $N_c - 1$ threads each receive subtasks, where N_c - number of cores and the sizes of subtasks are calculated using:

$$N_{sub} = \lfloor \frac{N_t}{N_c} \rfloor \tag{4.1}$$

where N_{sub} - number of tasks in a subtask, N_t - total number of tasks, N_c - number of cores.

The last thread receives the remainder of tasks which is always less than N_{sub}.

There are techniques that can help with avoiding the issue of frequent thread spawning and destruction. One of these techniques is the *thread pool*, a performant implementation is described in (Reinders, 2007). The main principle in a *thread pool* is that the threads for processing a large number of tasks can be preallocated and stored in an array (pool). The arbitrary number of tasks is then *scheduled* to a queue. Each thread from the *pool* then *pops* a task and processes it; after execution the thread is not destroyed, but it continues processing other tasks from the queue.

Both of the parallelelization implementations are implemented in `Parallel.h`.

4.2 GPU based implementation of TLED FEM with contact

4.2.1 Overview

The theoretical descriptions and justifications for the TLED approach and the Projection Based contact method are given in Sections 3.2 and 3.3 respectively. This section is dedicated for describing the details of their implementation.

The approach that needs to be followed when implementing these FE and

contact methods on a GPU is considerably different from the approach when a CPU only based implementation is undertaken. These technical issues and their solutions are presented in the following subsections.

The main aim of this section is to explain the implementation details that are concerned with GPU acceleration of the TLED FE and contact methods. It is thus important to provide a general introduction of all the essential details for general purpose computation on a GPU (GPGPU).

4.2.2 General Purpose Computation on GPU

GPU computational power

The graphical processing units (GPU's) have seen a major advancement in processing power due the demand of high quality graphics in computer games. In order to generate high quality visuals for computer games it is required to perform large amounts of calculations while keeping the calculation time very short. The traditional threshold for the minimal frame rate at which a series of images are perceived by an average human as a continuous motion is 24 frames per second (fps), known as the critical fusion frequency (CFF). This standard is mostly applicable to the TV and cinema industry. This is as opposed to gaming where frame rates of 60 fps and more are most desirable. Such high frame rates directly result in very short time budget for calculations for each frame. The 60 fps frame rate has a frame step time of:

$$\Delta t = \frac{1sec}{60fps} = \approx 0.01666sec = 16.66ms \tag{4.2}$$

where Δt - frame step time, fps - frames per second; the unit of the frame rate.
.

The large number of calculations required for realistic visuals are subdivided into distinct tasks. Most of these tasks are required to be performed within the time of a single frame step. The short time step implies that each of the tasks gets a shorter time window in which all its calculations need to be performed.

Additionally, most of the calculation tasks in a typical computer game will have a repetitive nature, whereby a large amount of similar calculations need to

be performed many times. Most of these calculations also have no dependency on others, which makes them ideal candidates for parallel processing. Such tasks are often called parallelizable.

The parallel processing of all the suitable calculations is what the GPUs are most efficient at. The architecture of the modern GPUs is focused on combining a large number of parallel processing units. Each of these units are dedicated to performing a part of the total amount of work. The independence of parts allows running the parallel units without waiting for other tasks to finish.

The tasks required for FEM calculations are similarly largely parallelizable. The tasks where a large number of elements need to be processed to identify the stress tensors within them are an example of such a task. The processing of nodes to perform time integration to evolve their positions is another example. A single set of calculations required for processing a single element or a node we will denote as a sub-task. The total number of sub-tasks will be shared between the number of available parallel processing units contributing to a unit's workload. Each unit will process the workload in parallel independent of the other units.

Before the introduction of GPUs the numerical methods traditionally utilized for the FEA implementation were implemented on CPUs. The difference between the CPU and GPU architectures, however, complicate the porting of these methods from CPU to GPU. GPUs use the single instruction multiple thread (SIMT) architecture, which places particular constraints on both the design and implementation of algorithms and data structures. This makes the porting of strategies often difficult, inefficient, or even impossible (Göddeke, 2011).

The workload distribution described above needs to be done with care. The application programming interface (API) for GPU programming should allow wide flexibility and low effort when implementing FEM on a GPU. The implementation should also be portable to different GPUs and hardware architectures.

Graphics Pipeline

The original and primary purpose of the GPUs is processing graphics. The processing is typically organized into a pipeline, initially introduced by Silicon Graphics Inc. (SGI).

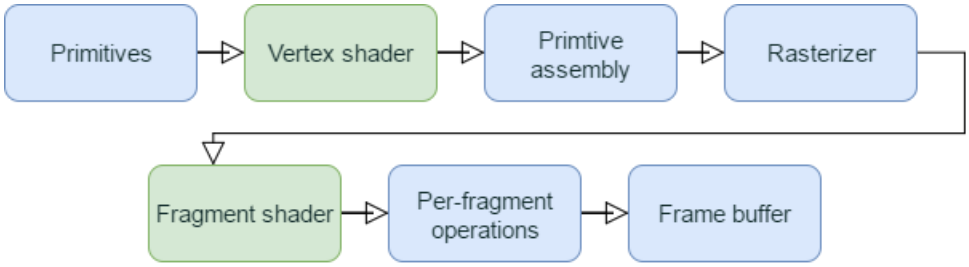


Figure 4.4: A basic graphics pipeline consisting of several stages. The vertex and the fragment stages are typically programmable and are used for GPU computation.

The pipeline is divided into separate stages. Each stage plays a specific role in obtaining the final rendering. Some of the stages have dependency on the previous stages. A very basic representation of a graphics pipeline is shown in Figure 4.4

The focus on graphics rendering implies that the hardware and software architectures are fine-tuned for graphics rendering. The types of calculations and the framework according to which the calculations are organized differs considerably from the traditional graphics rendering. This introduces considerable complications when implementing a solution for tasks different from graphics rendering.

The writing of the calculation results is particularly difficult due to the characteristics of the pipeline and the of the hardware (Göddeke, 2011). In a standard canonical graphics pipeline the write operations are only allowed to the currently bound render buffer in a form of a texture. Therefore, the whole program needs to be built with this in mind. A typical program that executes vertex or fragment operations for a certain visualisation task is called a shader. Shaders use a C-style syntax but are difficult to verify prior to execution as they are not compiled as part of the main application program which calls them.

High-Level GPU programming

The demand for using GPUs for general calculations resulted in the occurrence of several dedicated APIs. General purpose GPU (GPGPU) programming is the term used when describing this type of programming.

CUDA CUDA is a proprietary NVIDIA [1] GPGPU parallelization API. It is a popular choice in many areas including scientific research. This is due the fact that it provides a simple interface which avoids having to write the aforementioned shaders. Like C++ AMP it also provides a much simpler integration with the host machine code. The parallel kernels can be written inside C++ code using the `__global__` keyword. The ease of use makes developing parallel applications simpler and improves productivity.

High accuracy and performance make it appealing as the target parallelization library. However, the proprietary nature of CUDA makes it less attractive as it implies the non-optional requirement of having to use an NVIDIA GPU. The software written using CUDA can only be executed on GPU, as opposed to also being able to run on CPU in parallel or through FPGA's.

OpenACC The OpenACC standard [2] defines a common way of converting CPU targeted code to a format executable on GPU. The standard is similar to the OpenMP [3] API in that it is a purely `pragma` annotation based syntax. The implementation details of the GPU parallelization is hidden from the end-user. Instead the user is able to indicate the areas of the code that need to be parallelized using special annotations.

The advantage is that pragma based annotations can be added to the existing code executed on the CPU, which will then automatically execute on the GPU with improved performance. The intricate details of the GPU API's are hidden from the developer and the developer productivity is increased considerably.

While the solution eases the process of GPU paralellization, in using such a framework the user loses fine grain control. The trade-off may lead to less than optimal utilization of the GPU and have little facilities for manual tweaking of the underlying architecture.

OpenCL OpenCL [4] is an open-source cross-platform parallel processing API. The API is similar to OpenGL and provides low-level access to the hardware. It

[1] http://www.nvidia.com/object/cuda_home_new.html
[2] http://www.openacc.org/
[3] http://www.openmp.org/
[4] https://www.khronos.org/opencl/

is different from the other API's discussed above. It does not allow embedding kernel code inside. This, however, can be fixed by using third-party tools (Lawlor, 2003) that extract and compile OpenCL kernels from the host code. It is however similar to C++ AMP in terms of following a heterogeneous parallel computation model (Stone et al., 2010b).

Due to the fact that it is a lower-level heterogeneous API, OpenCL has certain drawbacks. The fact that the parallel kernels need to run on the wide range of supported hardware, implies that all OpenCL code needs to be loaded dynamically and precompiled on application startup.

After considering the drawbacks and the advantages of all the listed GPU parallelization approaches, the GPGPU approach chosen for the childbirth simulation system is based on the OpenCL language standard. The major advantage of using OpenCL is in its wide support on different platforms (OS'es) and hardware (CPUs, GPUs, FPGAs and coprocessors such as Intel Xeon Phi [1]). It should be noted, however, that the support for latest OpenCL features differs from one vendor to another. Namely, AMD has support for the OpenCL 2.0 standard whereas NVIDIA currently only supports the older OpenCL 1.2 standard. For the purposes of this project the older OpenCL 1.2 standard is sufficient.

GPGPU with OpenCL

OpenCL is based around the notion of *kernels*. Each *kernel* can be thought of as a unit of computation that will be performed multiple times in parallel. While each kernel is running in parallel it will be allocated an id.

It is important that the id is not arbitrary, but rather it is closely related to memory access. The nuances of memory access are discussed further in Section 4.2.9

As memory is one of the most important resources of the GPU, it is important to understand the memory architecture on the hardware level. In order to minimize the memory access overhead the memory architecture of the GPU needs to be taken into account as otherwise performance may suffer considerably.

[1] https://software.intel.com/en-us/articles/opencl-design-and-programming-guide-for-the-intel-xeon-phi-coprocessor

All modern processing units have a memory hierarchy. Typically the lowest level of memory hierarchy is the global omni-accessible shared memory. The highest in the hierarchy is the private memory accessibly only to the core that it belongs to. OpenCL API provides a specific way of using the GPU memory hierarchy (Munshi et al., 2011) as seen in Figure 4.5. The private memory provides small but very fast memory. The global memory is accessible from all threads, but carries large access time overheads.

GPGPU and Performance Issues

As already mentioned GPUs can provide a much higher performance as compared to CPUs. This, however, requires carefully choosing the implementation approach to ensure that the hardware is utilized in the most optimal manner (Eberly, 2014). There are also many other aspects of GPGPU programming worth considering, but these are most relevant:

- No locking operations
- Data layout and access patterns
- Branch-less code

One of the most important requirements is that there should be little and preferably no inter-dependencies between the calculation tasks executed by the GPU. The GPGPU APIs allow blocking operations whereby the whole GPU or a part of the processing units of it halts operation in order to wait for another processing unit to finish calculations. The use of such blocking operations should be avoided to keep the hardware from idling, when it could be performing useful work. Minimizing idle time is the primary way of increasing processing speed for a given GPU.

Another very important requirement for better performance is the data layout. The GPUs are designed to have fast access when a specific data layout and data access patterns are used. The preferred access patterns are largely localised, whereby the read and write operations are most optimized when they are performed within a specific region in memory. The memory architecture depicted in Figure 4.5 shows the layout.

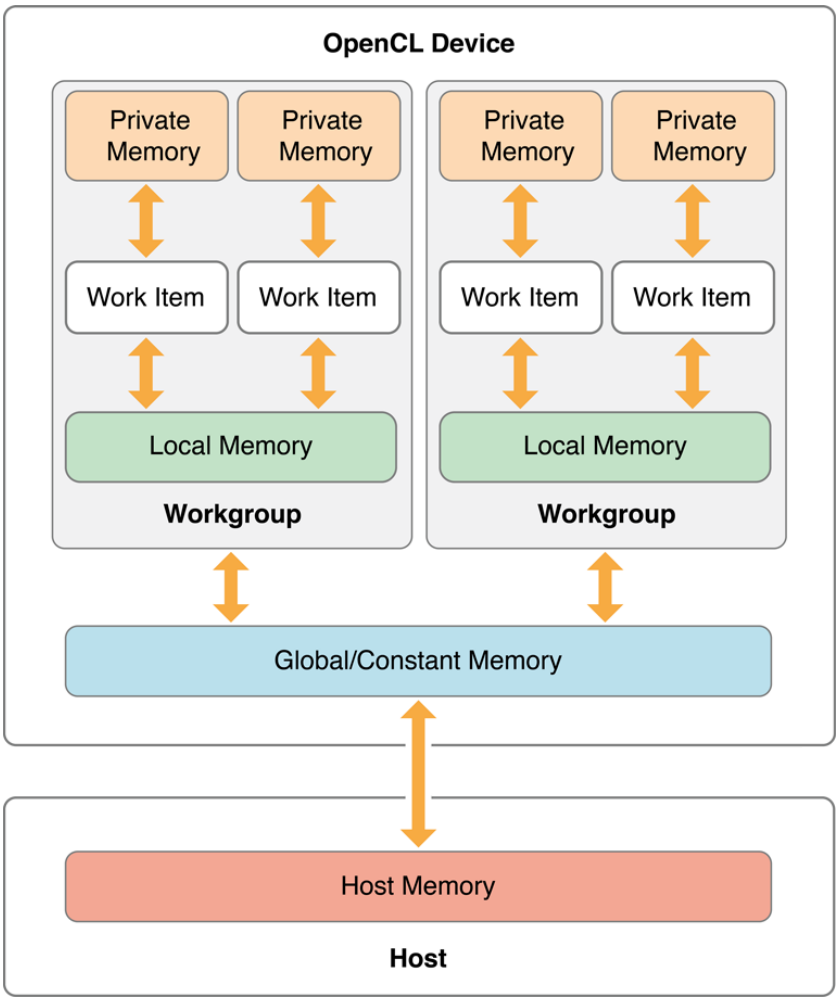


Figure 4.5: OpenCL memory model. The private memory provides small but very fast memory. The global memory is accessible from all threads, but carries large access time overheads. Image source Apple Inc (2013)

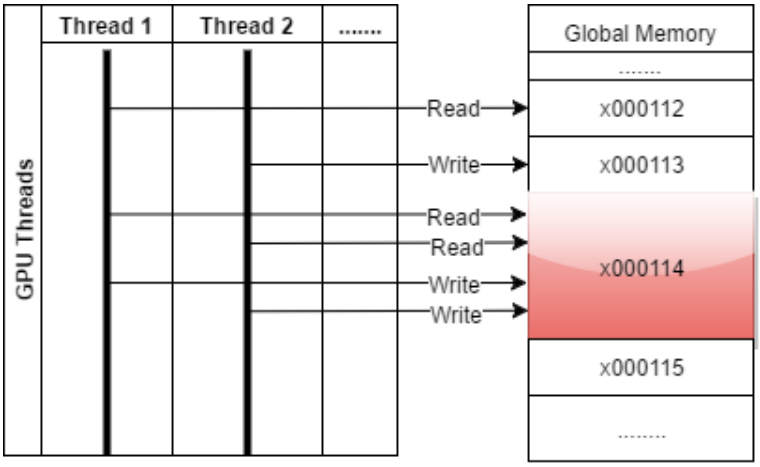


Figure 4.6: Data race condition in a parallel GPU application when two or more threads are attempting to read and write to a memory location. A data race may occur and the resulting value of the accessed memory location (red) will be undefined (impossible to predict).

The write operations should not be read-and-write especially if there could be more than one thread performing the operation as this could result in a data race. Data race may occur when two parallel threads executing simultaneously require a read-write operation. For instance, a kernel where a "+=" operation is performed on a global memory region may cause a data race. The result of the data race will be manifested as an erroneous calculation result stored in the addressed global memory location.

GPGPU programming also tends to prefer a branch-less code structure. Branching in code introduces a divergence between different processing units. An example of such idling would be in the case where a set of tasks needs to be performed and some of them are considerably longer than the rest. The processing units that have little work will finish and wait for the working units to finish.

Further details on the performance optimization methods with specifics relevant to the FE childbirth simulation are given in Section. 4.2.9.

4.2.3 TLED algorithm implementation

The algorithm can be divided into 3 distinct steps:

1. Precomputation
2. Initialization
3. Time stepping

Note how the precomputation step is a separate entry which allows excluding the computational effort from the main update loop. The values that can be precomputed are denoted using the notation from Bathe (1995). The 0 on the top left side of the symbols indicate that the value is for the initial reference configuration which stays constant throughout the simulation.

Here we provide a more detailed break-down of the steps, which are based on the description provided by Taylor et al. (2008):

1. Pre-computation
   (a) Read the input file and load the mesh along with other assembly information (constraints, external loads, etc).
   (b) For each element of the mesh compute the following quantities
      - Jacobian determinant $\det(\mathbf{J})$
      - Strain-displacement matrices $\boldsymbol{\partial h}_0$
   (c) Compute the diagonalized (lumped) mass matrix M_0 representing the mass of the whole mesh
2. Initialization
   - Set the nodal displacements $\mathbf{u}$ to the prescribed values and add prescribed external force loads into $\mathbf{R}$
3. Time-stepping
   - For each element:
     (a) Compute deformation gradient $\boldsymbol{F}$ based on the current nodal deformations u and strain-displacement matrix $\boldsymbol{\partial h}_0$ according to

$$\boldsymbol{F} = \boldsymbol{u}\boldsymbol{\partial h}_0 + I \tag{4.3}$$

(b) Compute Right Cauchy-Green deformation tensor $\boldsymbol{C}$

$$\boldsymbol{C} = \boldsymbol{F}^T \boldsymbol{F} \tag{4.4}$$

(c) Compute the Second Piola-Kirchoff stress matrix $\mathbf{S}$ from $\boldsymbol{C}$ and $\boldsymbol{F}$. The components of the matrix are found according to equation from (Taylor et al., 2008):

$$S_{ij} = \mu(\delta_{ij} - C_{ij}^{-1}) + \lambda \det(F)(1 - \det(F))C_{ij}^{-1} \tag{4.5}$$

where μ and λ - Lamè constants, δ_{ij} - Kronecker's delta.

(d) Compute reaction forces for the nodes of the current element. An efficient way of evaluating the integral in the equation is using a Gaussian quadrature according to Eq. 3.15

$$f = V_0 \boldsymbol{\partial h S F}^T \tag{4.6}$$

- For each node:

  (a) Update the deformation of the current node using a time-integration method (e.g. Central Difference Method)

  (b) Apply the boundary conditions for nodal displacements $\mathbf{u}$ including the plane projection constraints and add current external force loads into $\mathbf{R}$

The full algorithm can be visualized as a continuous cycle of data exchange between two mutually dependent functions. Figure 4.7 provides an intuitive visualization of the main principle of the TLED algorithm. The initial constant value pre-computation step is executed once, but the next steps are repeated within a loop. Within the cycle, the element stress calculation step is performed first. It calculates the second Piola-Kirschoff S_{PK} stress. The stress is used to calculate the contributions to the internal force of all the nodes connected to the element within the same step. The next step uses the nodal force contributions to combine them into a full internal nodal force for each node. The nodal force is then used to find the nodal acceleration and then integrate the acceleration to find the

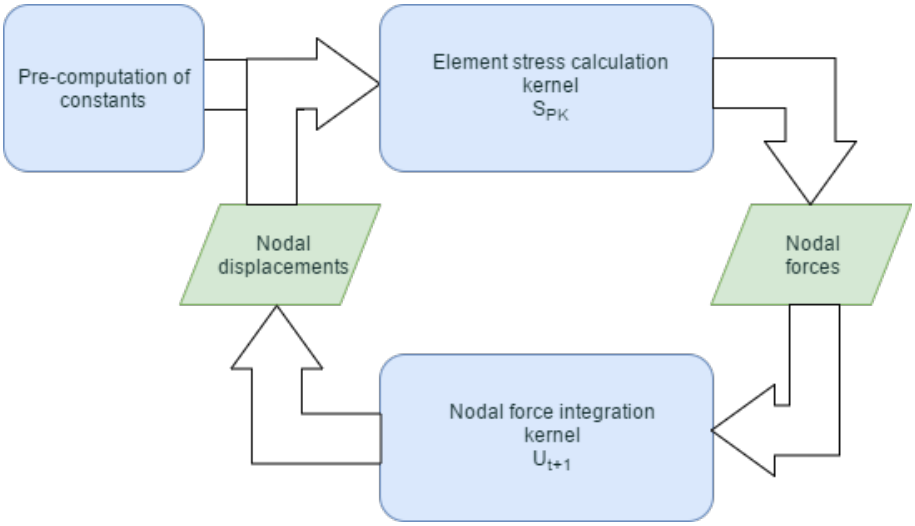


Figure 4.7: The core steps of the TLED algorithm. The main principle is that the two kernels are in constant exchange of data.

next nodal displacements according to the chosen time integration method. The new nodal displacements are fed back into the element stress calculation and the cycle continues. Each of the steps is represented by a separate kernel.

4.2.4 Pre-computation of constant values

The pre-computation step is concerned with calculating the following values:

1. Element volumes according to Eq. 3.26
2. Node masses based on element volumes and material density according as a product of the two
3. Shape function derivative according to Eq. 3.30
4. Element node indices based on element connectivity

The pre-computation of all the listed values is performed on the CPU. The computed values are then uploaded to the GPU global memory and kept there for further use. It is relatively slow when compared to the GPU calculations. The calculations are performed only once, so the performance of the pre-computation is less relevant.

4.2.5 Element processing

The computations that are concerned with calculating the stress occurring within a finite element are performed separately from the calculation of the per-node quantities. The description of the element processing is given in the following algorithm, which was implemented as an OpenCL kernel.

Algorithm 4 Algorithm for element processing

1: **for all** elements in assembly **do**
2: Collect nodal displacements into `float u[4][3];` using the per element nodal indices list from `nodeInds`
3: Extract the shape function derivatives `DhDx` for element from the global buffer `m_DhDx`
4: Multiply `u` with `DhDx` and add 1 to all three diagonal values of the product to calculate `F`
5: Calculate SPK according to the chosen material model. Neo-Hookean model is used in this implementation.
6: Calculate the nodal force matrix `Fe` using the SPK value
7: Spread the values from `Fe` to the global buffer `Fx` using the nodal indices `nodeInds`
8: Calculate the Von-Mises stress and stores it to the `m_VMS` buffer according to Equation 4.7.
9: **end for**

$$S_{VM} = \sqrt{\frac{(S_{11} - S_{22})^2 + (S_{22} - S_{33})^2 + (S_{33} - S_{11})^2 + 6(S_{12}^2 + S_{23}^2 + S_{31}^2)}{2}} \quad (4.7)$$

where S_{ij} - are components of the second Piola-Kirchoff stress tensor.

The calculated stress values in Equation 4.7 are useful when evaluating the accuracy of the TLED implementation in Section 5.2. They can also be useful in the future to indicate the areas of high tissue damage risk during training and predictive simulations.

4.2.6 Node processing

Time integration

Algorithm 5 Algorithm for element processing

1: **for all** nodes in assembly **do**
2: This is required for the reasons described in Section 4.2.10 the total internal nodal force needs to be summed from individual element contributions.
3: After the total internal nodal force is calculated, Equation 3.43 is used to calculate the next displacement of the node.
4: The next displacement is calculated and is stored in a buffer for the subsequent displacements `m_U_new`. The precomputed values stored in `m_ABC` are used to calculate the new displacement.
5: **end for**

Later, a swap operation is performed that copies the values from the `m_U_new` to `m_U` and the values from `m_U` are copied to `m_U_old`.

Boundary condition application

The boundary conditions are applied as part of the node processing stage. This is allowed as the BCs are restricted to be per-node, as opposed to allowing multiple BCs per node.

A detailed description of how the BC application is implemented is provided in Section 4.2.8

4.2.7 Data buffer layout

The OOP based implementations of the FEM tends to store element and nodal data in encapsulated structures or classes, whereby all the information pertaining to a given element or node are stored adjacently in memory. The alternative approach to the encapsulation based memory layout is sometimes named as *arrays of structs* layout.

Figure 4.8 gives a basic overview of the data-structure framework used in our TLED implementation.

o The data describing the actual finite element meshes comprising the simulation are stored in the `Assembly` class. The assembly can be thought of as the *model* in the Model View Controller (MVC) pattern as it has no indication of how the analysis should be performed.

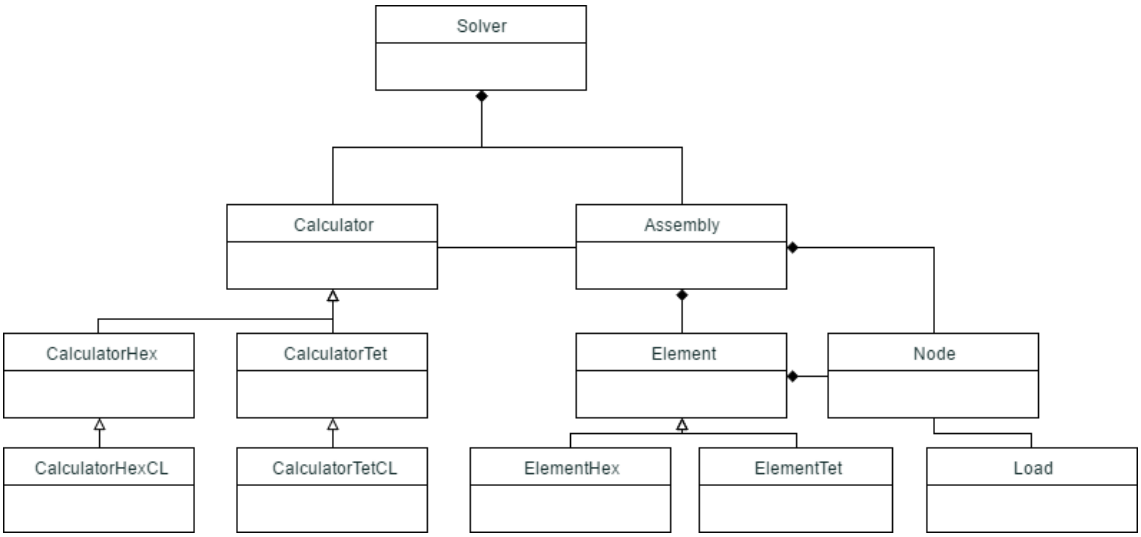


Figure 4.8: Basic class diagram of the TLED implementation. All computations are performed in the `Calculator` subclasses and FE data is stored in the `Assembly` class. operations

All the components of the system responsible for performing the analysis are stored in the `Calculator` subclasses. The `CalculatorTet` and `CalculatorHex` classes share the interface of their base class, but perform different operations. Further subclasses like `CalculatorTetCL` serve to extend the functionality of the base class which only executes the analysis on the CPU by running the analysis on the GPU using OpenCL. The pre-computation of the constant quantities is inherited and the actual update method is overridden to use OpenCL. Notably, the system could be improved by further separating the CPU and GPU relevant parts of the calculator into separate classes to implement a bridge pattern, which would result in future extensibility to support other types of execution environments and element types.

The composition relationship indicated by the black diamond are used to indicate that the assembly cannot exist without elements and elements do not exist outside of an assembly. The same applies to the instances of class `Node` that represent the nodes of the FE mesh.

Dynamic buffers

The buffers used as input for the TLED pipeline at each update are

- External force buffer - stores the values for the external forces acting on each of the nodes, 4 `float`s per node.
- Nodal displacement buffer - stores the current displacement values for each of the nodes
- Constraint type buffer - stores the type of the constraint currently applied to each node
- Constraint magnitude buffer - stores the magnitude

All the nodal buffers contain a value per node and have a size proportional to the number of nodes in the assembly. Element buffers store values per element in a similar manner. Note that most 3D force vector values are represented by 4 floating point values. This is due to the performance considerations described in Section 4.2.9 dedicated to TLED implementation performance. Aligned memory access leads to better cache utilization and gives better performance.

Constant buffers

The constant data is stored in constant buffers and is uploaded to the device once at the time of the application initialization.

- Element volume buffer - stores volumes of all the elements. 1 `float` per element
- Nodal mass buffer - stores masses of all the nodes - 1 `float` per node
- Shape function derivative buffer - stores the shape function derivative matrices for each element - 24 `float`s per element
- Element node indices - 4 `int`s per element

4.2.8 Boundary condition implementation

The boundary conditions are represented by two separate buffers. Both buffers have sizes equal to the number of nodes in the assembly. This implies that each node can have at most a single boundary condition.

The first buffer is called *nodeConstraintFlags_CL* and represents the type of the constraint. The constraints kernel is executed once per each node. The result of the `get_global_id` function of the thread represents the index of the node currently processed. The node index is used to index into the `nodeConstraintFlags_CL` kernel and find the type of the constraint applied to the node. The actual values that represent the constraints are stored in the `m_constraintMagnitude` buffer. Each of the values in the latter buffer are represented by a set of four `float`s. The actual physical interpretation of these values is different based on the type of the constraint.

The issue with the approach is that only a single BC can be applied to a node. This may present problems in more complicated simulation scenarios where more than one BC needs to be added to a single node.

There are currently several types of constraints implemented:

1. None - free node
2. Fix - disallow any displacement of the constrained node
3. Displace - prescribe a specific displacement on the constrained node
4. Linear - gradually move the constrained node to the prescribed position based on the current sub-step
5. Plane - project the constrained node onto a plane
6. Spring - a force is applied to the constrained node to keep it at the required attachment position.

The presence of the *none* boundary condition type is required as the per-node layout of the BC buffers implies that every node always has a BC ascribed to it. However, not all nodes will have active BCs on them. Therefore, most of the

nodes will have the *none* type flag set on them indicating that there are no active BCs on the node.

The *spring* BC type can be interpreted as a basic linear spring model, which allows the constrained nodes to be fixed at a position with a degree of tolerance. The first three components of the 4 component value from the constraint magnitude buffer are used as the spring attachment point coordinates to which the constrained node will tend to be attracted. The stiffness of the spring is specified as the fourth component of the constraint magnitude. The actual constraint is executed as a basic Hookean spring that applies a force linearly proportional to the distance between the current position of the constrained node to the spring attachment point.

The *plane* constraint is the main constraint type used with our projection based contact method. A node that has a plane constraint attached to it will be projected onto the surface of the plane described by the four `float` values stored in the `m_constraintMagnitude` buffer. The first 3 values represent the normal vector of the plane and the fourth represents the distance to the origin. The algorithm used for the projection can be summarized as follows:

Algorithm 6 Projection constraint enforcement algorithm

1: Retrieve current node displacement u
```
float3 u = m_U_new[nodeIndex].xyz;
```
2: Retrieve constraint plane equation
```
float4 plane = m_constraintMagnitude[nodeIndex];
```
3: Calculate dot product with plane normal
```
float d = dot(u, plane.xyz);
```
4: Find node penetration into the plane's negative half-space
```
float penetration = plane.w - d;
```
5: **if** penetration is greater than zero **then**
6: Find projection onto the plane surface
```
float3 proj = u + plane.xyz * penetration;
```
7: Prescribe new nodal position to be on the surface
```
m_U_new[nodeIndex] = (float4)(proj, 0.0f);
```
8: **end if**

The *plane* constraint type is used for the projection based contact. As the plane equation value changes the node will be always on the positive half-space

or the surface of the plane. In the case when the penetration is negative, which implies that the node is away from the plane and is not penetrating, no influence is applied to the node. This allows it to behave normally as a free node in the FE mesh even when the boundary condition is applied. The conditional in the code implies divergence in terms of parallel execution on the GPU, but no measurements are performed.

4.2.9 OpenCL performance considerations

The use of GPGPU programming requires additional considerations when implementing GPU based TLED. The nature of the TLED algorithm adapted from Miller et al. (2007) is readily paralellizable in the sense that the tasks of computing the element stress tensors and nodal time integration can be run in parallel without major change in the basic algorithms. However, there are still aspects of the implementation that need to be considered in order to maintain high performance.

Synchronization-Free Cooperation

The type of synchronization commonly used in GPU programming is typically referred to as the synchronization barriers. When used excessively these barriers can degrade performance. Such cases of degraded performance are particularly apparent when not all work-items are required to participate in the cooperation. This implies that part of the work-items are idling. The wavefronts running within a single work-group are able to perform their operations independently of each other; when the number of work-items is more than the size of the wavefront and the synchronization barriers are utilized they fail to do so.

Apart from synchronization barriers there are also atomics that serve a similar purpose of preventing simultaneous read-write operations on a single chunk of memory. Using atomic operations is considerably more efficient than synchronization barriers, but yet incur certain overhead, which will be worse than the synchronization-free cooperation.

The TLED kernels implemented in BirthView are designed to avoid barriers in order to maintain synchronization-free cooperation. The TLED kernel for per

element nodal force calculation is an example of a situation where synchronization barriers could be utilized. This is motivated by the fact that the individual nodal force contributions from a single element need to be added up into a single nodal force. Threads running in parallel may end up writing to the same location simultaneously thus causing a data race. Our implementation, however, avoids data races without using barriers. The approach used to achieve this is based on spreading the write operations to a larger buffer of forces which can then be gathered when the total forces are required. This approach is based on the implementation by Taylor et al. (2008) in their GPU TLED implementation using CUDA. More technical detail on the way this technique was implemented is provided in Sec. 4.2.10.

Loop unrolling

Typical loops are executed in the way that assembly level jumps are required and a loop variable needs to be compared in the loop condition. It is possible to avoid the extra overhead of these operations by *unrolling* them. Unrolled either through a compiler directive or manually by using a tiered function hierarchy.

The OpenCL compiler will attempt to perform automatic loop unrolling where possible. The unrolling will be performed only if the total number of iterations is known at compile time. The most performance critical loops in our TLED kernels are a part of the matrix multiplication functions. These loops have a strictly defined number of iterations, which allow the compiler to unroll them implicitly. The OpenCL compiler also allows using compiler directives kernels for explicit loop unrolling (Stone et al., 2010a).

Maintaining high occupancy

It can be seen that most of the above methods for increasing the performance are concerned about the idle time spent by the GPU compute units. The measure of how little the compute units are idling is referred to as the *occupancy*. Following the principles described above, can result in better occupancy levels and thus better performance.

Increasing the occupancy of the GPU directly translates to an increase in per-

formance. It should be noted, however, that occupancy is not the only parameter required for optimal performance.

Coalescing memory access

The performance of memory access is strongly sensitive to the order in which the global memory is accessed. Any memory access on a GPU is performed in chunks (Stone et al., 2010a). The memory adjacent to the requested data has a high chance of being fetched along with it. When adjacent work-items also access adjacent memory regions there is good chance that GPU memory cache will allow the data to be shared efficiently without unnecessary global memory access. Any algorithm that could take advantage of the pre-fetched data would be able to maintain better efficiency.

The TLED GPU kernels attempt to keep the memory accesses coalesced where possible. However, the tetrahedral FE meshes used are highly unstructured. When performing per element operations in a kernel there may be need in accessing per-node data. This is the case in the element force calculation kernel. The kernel is executed for each element, but the resulting calculated forces need to be distributed to each node the element is connected to. The resulting order in which the nodes are addressed is not sequential, which implies that data from widely ranging memory regions needs to be accessed. This is an example of uncoalesced memory access and incurs considerable performance overhead.

Optimal use of GPU registers

The actual operations performed by the GPU compute units are performed on operands stored in registers. When an operation, such as an arithmetic operation, needs to be performed on data from global memory the data is loaded into registers. The compute units have a limited number of registers. Additional details about the GPU registers and their use in OpenCL can be found in (Munshi et al., 2011).

All of these registers are shared between all the compute units when performing computation tasks. Each of the threads running in parallel executes an instance of the current kernel. The number of vector registers used by a thread

is determined by the nature of the actual calculations performed by the kernel.

The GPU used for the experiments described in this thesis on an AMD Radeon 8970M only has 256 vector registers. The resulting GPU occupancy is therefore only 50%. It is very difficult to control the number of registers used by a kernel as the compiler from the AMD OpenCL driver produces assembly code using heuristics. When combined, these heuristics can produce unpredictable assembly code, which makes kernls hard to optimize (Soos, 2012).

Kernel Fusion

Kernel fusion is a technique whereby two or more kernels are combined into a single kernel. Whereas this does not change the total number of calculations required to execute the kernels, it does reduce the number of API calls and scheduling operations that are needed to be performed by the GPU driver.

Additionally, memory caching considerations are also relevant for the described case. In order to pass the calculation results from a task located in a single kernel to another task in a different kernel they must be saved in the global memory. This necessarily incurs additional memory write overhead. When two separate tasks are located within a single kernel the fast local memory can be used. This is opposed to exchanging data between two separate kernels through the global memory. This is illustrated in Figure 4.9.

The initial implementation of TLED GPU used 4 kernels in total. The extra kernel is for gathering the spread element nodal force contributions. As compared to the final solution where only 3 kernels were used, the extra kernel was executed between the element nodal force kernel and the nodal integration kernel. The performance impact of the kernel fusion was the decrease of the total TLED processing time from 58ms to 52ms.

Assembly fusion

Kernel fusion has been identified as the way to increase the performance. The same principle can be applied to the case of multiple assemblies as well. An assembly in the context of our simulation system is a combination of all the finite elements, nodes and the list of materials referenced by the elements. For instance

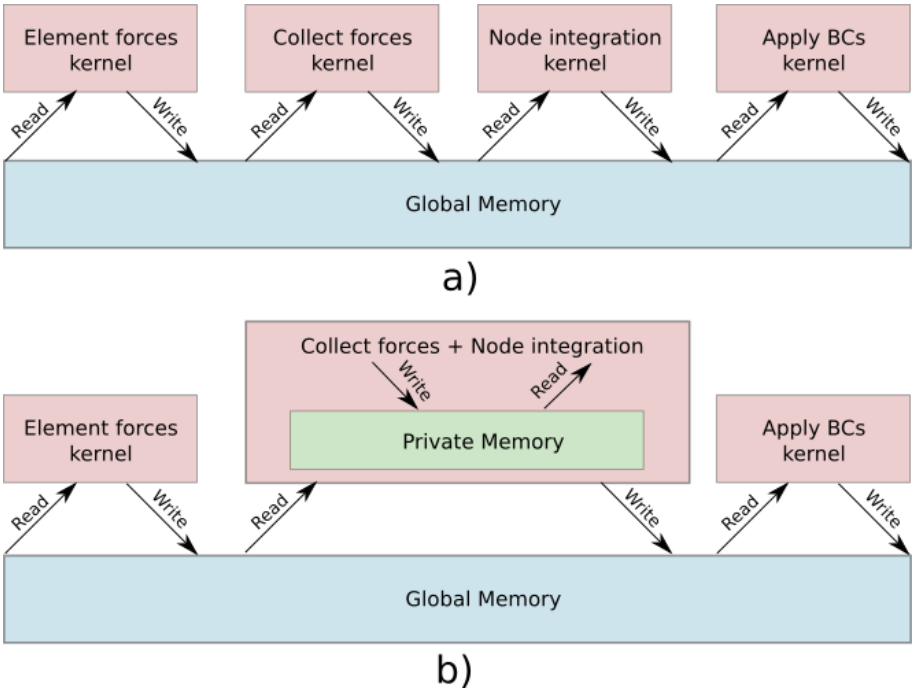


Figure 4.9: Visualization of kernel fusion. The initial configuration where the two kernels are separate (a) and fused kernels (b). Collect forces and node integration kernels were combined, which allowed avoiding unnecessary global memory access.

the cervical and pelvic floor FE models were treated as separate assemblies with all the FE data stored and treated separately. After combining the meshes it was found that the total TLED processing time dropped even further from 52ms to 30ms.

Each assembly is processed separately by invoking all the OpenCL API calls. This also implies that all the data transfers between the GPU global memory and RAM need to be performed separately for each assembly. Compared to the API calls overhead the data transfer seems to be the most impactful factor.

The overhead of having multiple assemblies is caused by factors similar to kernel fusion although the effects are more severe. Kernel fusion improved performance for about 10% whereas the assembly fusion allowed an improvement of approximately 42%.

Node valence

There are nodes that have more than one element contributing to its internal force. The number is referred to as the valence of the node. The kernel executed per node dedicated to the collection of the internal force contributions is responsible for accessing each element connected to a node. The number of connections (valence) of a node varies largely from node to node. The GPU threads processing the node with a high number of incident elements (valences) will take longer to complete than the ones with small valence. This introduces divergence in the GPU execution and reduces utilization. The utilization is reduced as a result of all the other threads within a wavefront waiting for the thread processing the larger number of valences. Therefore, the nodal valence should kept as small as possible and the variation between nodes should be reduced if possible.

4.2.10 Avoiding data races by using an extended nodal force buffer

In a single CPU implementation, the solver processes the constraints one by one in a Gauss-Seidel fashion. Thereby, after each constraint projection, the positions of affected particles are immediately updated. In a parallel implementation, the constraints are processed in parallel by multiple threads. If two constraints affect-

ing the same particle are handled by two different threads simultaneously, they are not allowed to immediately update the particle's position because writing to the same position simultaneously leads to race conditions making the process unpredictable.

A solution to circumvent this problem is to use atomic operations. Such operations are guaranteed to be exclusive. However, using atomics can slow down parallel execution significantly especially on GPUs as there is often need to synchronize the tasks running in parallel.

The issue can be addressed by an alternative approach if each element is allocated with a separate memory chunk to store the individual force contributions from each element separately. A buffer with sufficiently large size can be allocated to store these contributions separately.

The layout of the buffer can be visualized as seen in Figure 4.10. The three elements are actually connected and represent a cohesive assembly as indicated by the shared nodes. These nodes are connected to more than one element (nodes b, c, d, e). The GPU will be running three threads for each element. All three element threads may at some point attempt a read-write on node c as it is shared by all three elements. This may lead to a race condition. However, if there are three separate values (pink f_3, green f_2 and cyan f_1) no race condition will occur. None of this applies to non-shared nodes (a and f).

The trade-off for this approach is that the individual contributions need to be summed up at the point when the total nodal force is needed.

4.2.11 Comparing Performance of Gather and Spread operations in element and node kernels

The trade-off requiring the summation of the separated nodal contributions can have considerable effect on the performance. To investigate potential optimizations the memory access patterns should be studied.

There are two approaches that can be used for organizing the data storage in the extended internal force buffer. The difference between the approaches is in the order in which the nodal force contributions are stored in the extended buffer.

The first approach is to write the nodal force contributions from element

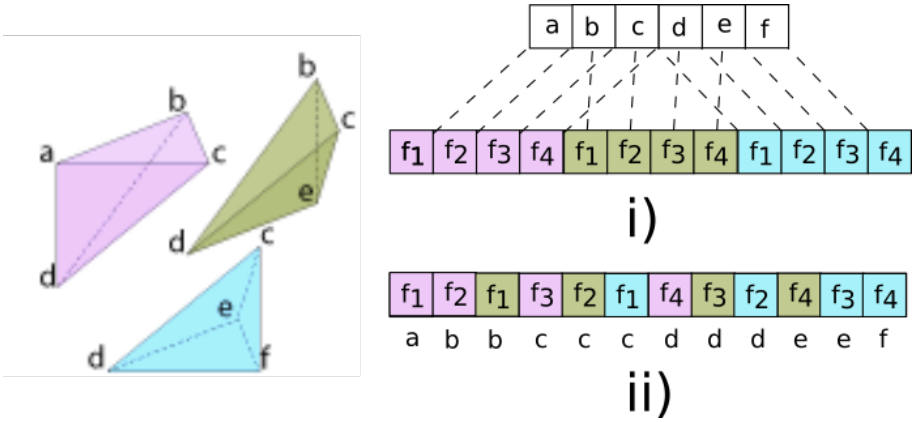


Figure 4.10: Force contributions for an assembly of 3 elements and 6 nodes. A larger buffer (12 values) is needed to store the individual force contributions from elements. The nodes and their corresponding contributions are visualized (i). Alternative ordering of the buffer to have node-coalesced access (ii).

kernels using a scatter write. Figure 4.10 (i) shows this memory layout. Note that from the element's perspective the memory is coalesced (the colours are common for consequent cells). This implies when element kernels are executed the GPU can write data sequentially. However, when nodal kernels are executed the data needs to be gathered from non-sequential memory locations.

The second approach is different in that it forces the element kernels to write into non-sequential data. That is the force data is written into sequential regions of force contributions coalesced with respect to nodes. Figure 4.10 (ii) shows this memory layout. With this approach the access from the perspective of nodes is sequential and will be performed considerably faster. However, element kernels will spend time to perform the scatter write operations, which could be slower.

Tests were performed to identify the effects on performance of adopting either of the approaches. The tests were performed on a FE mesh of the pelvic floor and sacrospinous ligaments. The number of elements in the mesh is almost 30k and the number of nodes approximately 7.5k.

We found that executing a step took 259.8 μsec when using the first approach. Whereas, utilizing the second approach took only 198.3 μsec. This results in approximately 30% performance difference when compared against the gather in CTD approach. The implementation of NiftySim could benefit from removing

the gather operation and executing spread in the element kernels.

4.2.12 Extended force buffer size optimization

An early version of our GPU implementation was using the largest valence values for each element. For a FE mesh the total number of elements in the extended buffer would be calculated according to $N_{el} \cdot max(V_n)$, where N_{el} - number of finite elements, $max(V_n)$ - highest nodal valence. For a mesh with 30k elements and the maximum nodal valence of 33 the number of elements becomes $30000 \cdot 33 = 9900000$. Considering that each force value is stored in a `float4` struct for memory alignment and it has a size of 4 floats 4 bytes each, the number of bytes required to store the whole buffer becomes 15.84 megabytes. The size is not excessively large when compared to the modern GPU memory volumes of several gigabytes, but the GPU is required to very rapidly access and modify that memory region. Memory locality is beneficial for GPU implementations as discussed earlier in this section.

The size of the extended force buffer can be reduced by packing the force values more densely. This implies using a separate buffer for storing the indices of where the force values are stored in the extended force buffer. For the same case where the total size of the extended force buffer was 15.84 megabytes, the size of the dense stored extended buffer becomes considerably less and equal to 1.208 megabytes.

4.2.13 Implementation of the Projection Based Contact method

The theoretical description of the projection based contact method developed for the purposes of the childbirth simulation is described in Sec. 3.3. The general algorithm describing the steps that need to be taken as part of the Projection Based Contact method is covered there. The mathematical justification for the assumptions made as part of the contact method are also provided there. This section is dedicated to describe the technical details of implementing the projection based contact model.

Generic collision detection

The general task of managing all collision detection in the simulation system is taken by the `CollisionSystem`. The collision system keeps track of all the entities that have `Collider` components attached to them. Any of these entities can start overlapping each other during the simulation and suitable collision resolution needs to be applied.

Any entity's collision shape is an approximation of the actual shape of the object when it interacts with other objects. These can be thought of as the bounding volumes of various shapes. The simulation system has the ability to simulation contact between the following shapes:

- Sphere
- AABB
- OBB
- Ray
- Triangle (single)
- Bitree
- Octree

The contact between each unique pair of these shapes is detected and resolved by the classes with names starting with `CollisionMethod`. For example contact between an AABB and a sphere can be managed by an object of class `CollisionMethodBoxSphere`. The objects of the `CollisionMethod` subclasses are also cached for the cases when there is need to access the contact state of the entities in the previous updates. The cashes are kept in thread safe manner allowing for access from concurrently executing systems in different threads.

There is also a layering system implemented to prune out all contact between objects between which contact needs not to be detected. This is implemented by populating a matrix of Boolean values. The column and row indices represent the layer numbers and the Boolean value at that position in the matrix specifies

if the objects from the two layers should have contact between each other when they start overlapping in the 3D scene. This is also useful for the cases when several objects from the same layer need not to interact with each other, but need to have simulated contact with any other objects in scene.

The part of contact detection responsible of finding the intersecting triangles is generic. It can be applied for collision detection between any other object in a scene that has a 3D mesh assigned to it. This part is described in the classes `CollisionMethodBitree` and `CollisionMethodOctree` based on the arity of the tree used for collision detection acceleration. The two classes are C++ functors in the sense that they are invocable using the standard round braces function invocation syntax. Upon invocation the objects of these classes will perform collision detection and return an object of `MeshContact` class. `MeshContact` class is a subclass of the `Contact` class and among other meta details, carries the list of pairs of triangles that are intersecting. It also provides to extract all the unique triangles from the list of pairs, as the same triangle can be intersecting several other triangles thus being listed in the pairs list several times.

[Figure showing the data transfer between the different classes with arrows pointing from one to another and annotations above arrows indicating the type of data transferred]

Contact between rigid and deformable objects

The ability to take part in contact resolution is ascribed to an entity by adding a `ContactComponent`. This component is different form the `ColliderComponent`, which is used to ascribe the ability to take part in contact detection and no resolution will take place for the entity. The addition of `ContactComponent` to an entity also requires that the entity has a TLED `SolverComponent` attached to it, as the former component only deals with contact between deformable and rigid simulated objects.

The component allows to choose between the Newman BC based contact and the Project Based contact methods. The contact detection and contact resolution is decoupled, which allows the dynamic switching described above to be implemented. Both of the specific methods are described by the `NewmanContactMethod`

and `ProjectionContactMethod` respectively. This part of the method responsible for applying the BCs that represent the contact conditions is specific to FE implementation and is thus kept in the specialized classes.

The entity representing the deformable object will be treated as a rigid body for the duration of the contact detection stage. This stage is aimed at identifying all the pairs of intersecting objects. The presence of the `ContactComponent` necessarily implies the presence of the `ColliderComponent` making the entity take part in contact detection. Upon a successful TLED displacement update, the latest displacement values are applied to the undeformed mesh to arrive at the latest configuration. This updated configuration is transferred to the `ColliderComponent` and treated as a rigid body as described in the dedicated Section 3.3.4.

Parallel contact detection with amortization

The systems responsible for the TLED implementation and the contact detection are disjoint and are running in parallel. The concurrent execution of the two systems is demonstrated, in Figure 4.3 from Section 4.1.3. The TLED task is typically the longest running among the parallel running tasks, but it consists of a number of subtasks representing TLED sub-steps. Each sub-step represents a complete TLED update, whereby the full algorithm described in Section 4.2.3 is executed. These sub-steps are considerably smaller than the main simulation time-step of 16ms.

The main principle of amortized contact detection is to perform contact once per frame as opposed to performing it every TLED sub-step. With the current implementation it is infeasible to run the simulation even remotely at real-time rates, when the amortization is reduced to have contact detection every TLED sub-step.

The aim of amortization is to avoid performing contact detection every single TLED sub-step update. The amortization is possible due to the slow rate of the simulated process, thus temporal coherence allows us to avoid performing the whole of contact detection and resolution on every update.

Additionally, the fact that contact detection is executed on the CPU intro-

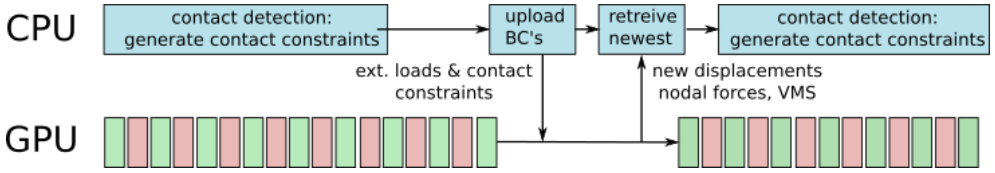


Figure 4.11: Contact detection executed on the CPU generates contact conditions. During *sync* the data is exchanged between CPU and GPU. The latest contact conditions are used by TLED until the next *sync* event and the latest deformations are considered constant until the next *sync* event in contact detection. Green and red sub-tasks depict per-element and per-node kernels running in succession on GPU.

duces major overheads for data transfer. The data transferred from CPU to GPU are the BCs representing the contact constraints generated as the result of the contact detection stage. The number of data transfers is greatly reduced when amortization is used. The data is only exchanged during the *sync* event described in Section 4.1.3.

The amortized contact detection does not imply amortized contact resolution in the form of contact condition enforcement. The contact conditions from less frequent contact updates can be enforced continuously as part of the more frequent TLED updates.

The amortized contact detection affects the simulation stability. The contact conditions can become obsolete if the TLED updates between the amortized contact updates lead to large deformations. The tolerable magnitude of the deformation between the updates can be calculated. For the case of the pelvic floor model with the average element extent of $0.5mm$ and the amortized time step of $16ms$ implies that the simulated motions should preferably be less than $31.25\frac{mm}{sec}$.

Chapter 5

Experiments and Results

5.1 Overview

Upon completing the GPU TLED implementation it became possible to perform FE simulations of soft-tissue. The implementation was applied to the simulation of the cardinal movements.

To ensure that the results of the simulations are correct, the implementation needs to be validated. The evaluation of the simulation starts with investigating the accuracy of the FE implementation. The results of these validation tests are presented first in this chapter. The next step of the evaluation is concerned with conducting experiments to investigate if the created simulation system is capable of achieving the main research aim of this thesis: simulating the cardinal movements of childbirth. The details of the said experimental setups are described, including the initial positions, orientation and physical properties of the objects taking part in the simulation. Then the simulated fetal head trajectory is analysed to check if the cardinal movements have been successfully simulated.

The real-time physics based simulation system allows performing series of experiments to investigate the cardinal movements of labour. The results of the experiments are then presented in a separate section and illustrated with screenshots of the various relevant stages of the simulation. The motions and rotations of the fetal head are of most interest and therefore are presented in graphs. The graphs demonstrate the changes in the position and rotation of the

head with respect to simulation time.

5.2 Accuracy and performance analysis

5.2.1 Assessing GPU TLED accuracy

Cube stretch in Abaqus and BirthView

A dedicated exporter class that allows exporting the FE model of an entity in the Abaqus CAE [1] compatible format was incorporated into BirthView. The class reads the FE mesh geometry generated by the TetGen module and generates a corresponding Abaqus .INP file with the same geometry and the material properties. The file can then be imported into Abaqus Explicit module and an analysis performed.

The validation test setup consisted of a simple cube that was stretched to varying degrees and varying directions. The dimensions of the cube were $1m \times 1m \times 1m$. The material density $1000kg/m^3$. A Neo-Hookean material model was applied to the cube with a shear modulus of 660kPa and a bulk modulus of 1MPa.

The Abaqus CAE requires converting these values to C10 and D1. The conversion formulas are shown in Equation 5.1 and were retrieved from the Abaqus User Manual (Hibbit et al., 2007).

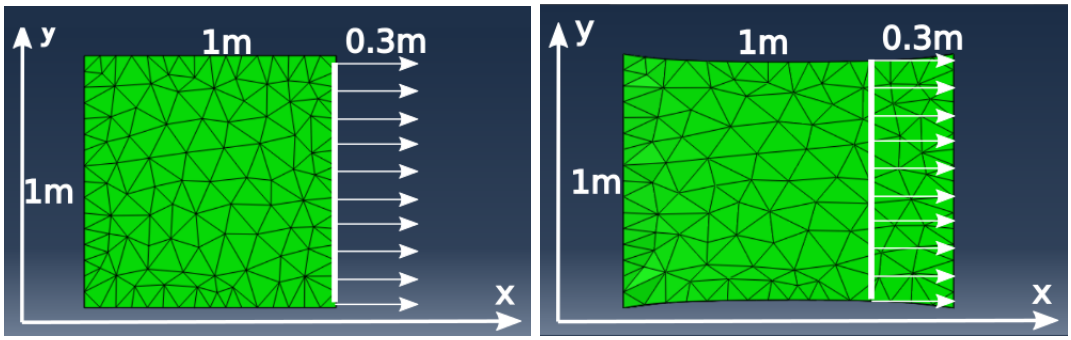
$$C_{10} = \frac{\mu}{2}, D_1 = \frac{2}{k} \tag{5.1}$$

where μ – shear modulus, k – bulk modulus.

All nodes on the side of the cube facing along the positive x axis were pulled by applying a linear type boundary condition. The magnitude of the stretch is equal to 30cm. The boundary condition is designed to gradually grow from 0 to full magnitude over time as in explicit dynamic analysis instant BC enforcement would result in a diverging solution. Figure 5.1 demonstrates the cube's configuration.

To replicate the same BC in the Abaqus Explicit software it was used with an *amplitude* assigned to it. The amplitude allows tabulating the values of the BC

[1] https://www.3ds.com/products-services/simulia/products/abaqus/abaqusexplicit/



(a) The view of the test cube before stretching. White arrows indicate the direction of stretching.

(b) Test cube after stretching by 30 cm in the positive x direction. The depth dimension is not visible.

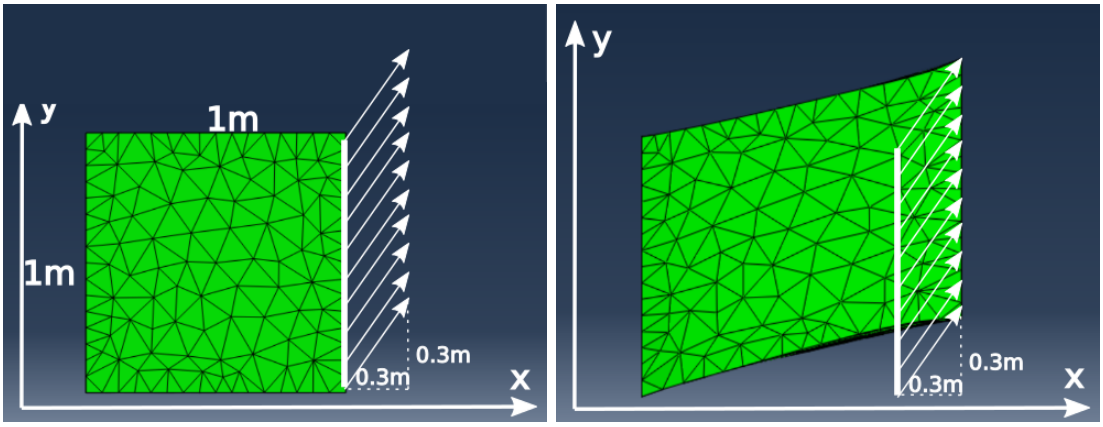
Figure 5.1: Cube uni-axial stretch validation test description.

from 0 to full magnitude depending on time. For both BirthView and Abaqus setups the duration of the gradual BC increase was set to 3 seconds, i.e. after 3 seconds of simulation the BC was enforced fully. The simulation was continued for an extra 5 seconds to ensure that all oscillations were dampened out to avoid erroneous readings.

Additional validation tests were performed with a similar setup to observe the validity of the results under a different stretch scenario. The same cube was stretched by 30cm in all three dimensions in the positive direction. That is each of the nodes from the side facing along positive x were displaced by a displacement vector $U = [0.3, 0.3, 0.3]$. Figure 5.2 illustrates the setup. Note that the stretch in the depth dimension is not visible.

A mesh comparison utility was developed in order to evaluate the differences between the results of the Abaqus simulation and the BirthView simulation. The utility was able to parse the (.RPT) report file generated by Abaqus upon successful analysis. The displacement values from the report file were then used to reconstruct the deformed mesh from Abaqus. The parsed displacements were then compared to the corresponding displacements reported by BirthView and the differences analyzed and visualized. The colour coding approach was used to display the areas where differences between the meshes is noticeable.

The results are reported in Table 5.1. It can be seen that the errors between the two simulation platforms are relatively small. The maximum relative



(a) The view of the test cube before stretching. Red arrows indicate the direction and magnitude of stretching.

(b) Test cube after stretching by 30 cm in the positive x, y and z directions. The depth dimension is not visible.

Figure 5.2: Cube multi-axial stretch validation test description.

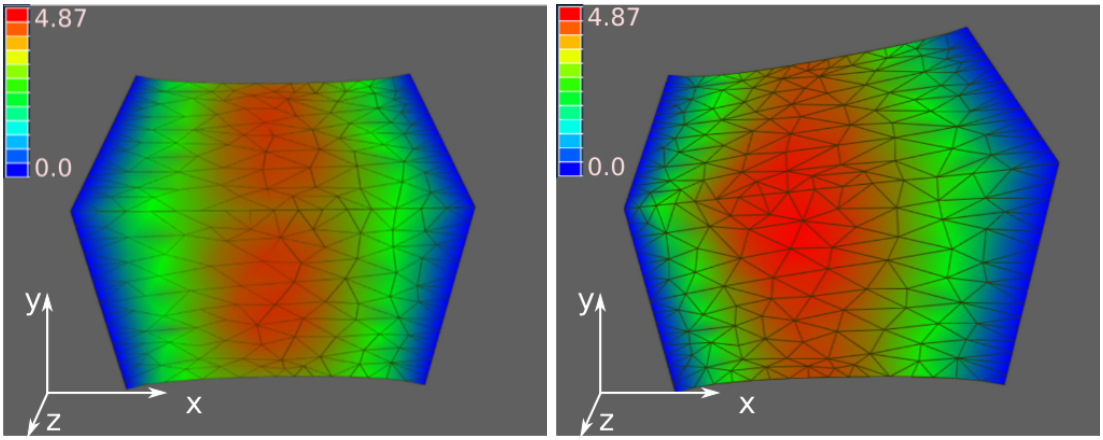
Table 5.1: The absolute and relative results of validation using the Abaqus Explicit

Direction	Magnitude	Max error	Max error	Avg error	Avg error
	mm	mm	%	mm	%
x	300	3.6	1.14	1.2	0.38
xyz	520	4.87	0.96	1.965	0.378

errors are under 1% as compared to the displacement applied. The errors averaged between all nodes are even smaller. The visualization of the displacement distribution is given in Figure. 5.3.

The Mises stresses were also compared to further investigate the differences between the results of BirthView GPU TLED and Abaqus Explicit. Table 5.2 provides the numerical values for the maximum Von Mises stress reported by Abaqus Explicit and BirthView.

It should be noted that the locations of the maximum stress and the distribution of stresses on the surface of the cube are largely similar. This is illustrated in Figures 5.4 and 5.5 showing the distribution of the Von Mises stress on the surface of the cube. The areas with the largest stress values are concentrated around the vertical band around the center of the cube. The same distribution



(a) (b)

Figure 5.3: Oblique view of the stretched cube. Colour coded visualization of the displacement error distribution is overlayed on the surface of the cube. The red areas indicate larger errors. (a) cube stretched in positive x, (b) cube stretched in x, y and z.

Table 5.2: Maximum Von Mises stress reported by Abaqus Explicit and BirthView.

Direction	Max Von-Mises stress, kPa	
	BirthView	Abaqus Explicit
uni-axial	300	465.3
multi-axial	640	963.4

Table 5.3: Cube compression results with varying compression weight for the projection based contact and the values generated with Abaqus Explicit.

	projection contact	Abaqus Explicit		
compression force, N	max U_2, mm	max U_2, mm	error, mm	relative error, %
98.2	5.91	5.794	0.01	1.72
185.6	11.12	11.124	0.04	0.35

appears on both results in a similar fashion. The small high stress areas near the upper vertices of the cube are only present in the Abaqus result.

5.2.2 Assessing projection based contact accuracy

In order to assess the accuracy of the proposed projection based contact method, an comparative experiment was run in both BirthView and Abaqus Explicit.

Comparison with Abaqus Explicit

A set of boundary conditions were introduced on the cube to replicate the same behaviour in Abaqus Explicit as in Birthview. All nodes from the top face of the cube were assigned loads replicating the pressure from a pressure plate. The force magnitude of the pressure load is equal to the weight of the block pushing downwards.

The results of the analyses are reported in Table 5.3 as the vertical displacement (compression) of the cube under the weight of the pressure plate.

The error of 1.72% is measured between the BirthView's projection based method and the Abaqus Explicit. This could potentially be the result of the the stress value errors observed in the previous experiment (Table 5.2), as the stress value differences will have an effect on the force resisting compression.

Varying mesh complexity

The effects of increasing the number of elements of the cube (from 127 to 21588) are presented here. This experiment is aimed to show the robustness of the contact method and its consistent behaviour. It was deemed beneficial to verify this assumption experimentally. Additionally, recall that the number of sub-steps

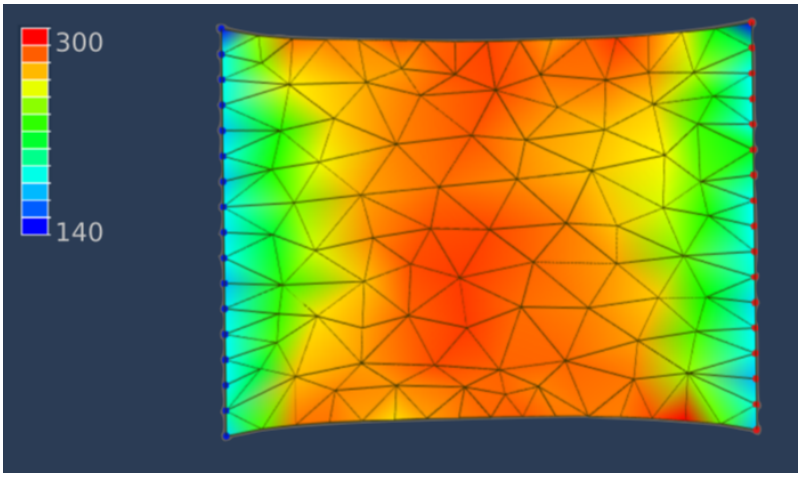


Figure 5.4: The view of the test cube after stretch in BirthView. Colour overlay shows the con-Mises stress distribution.

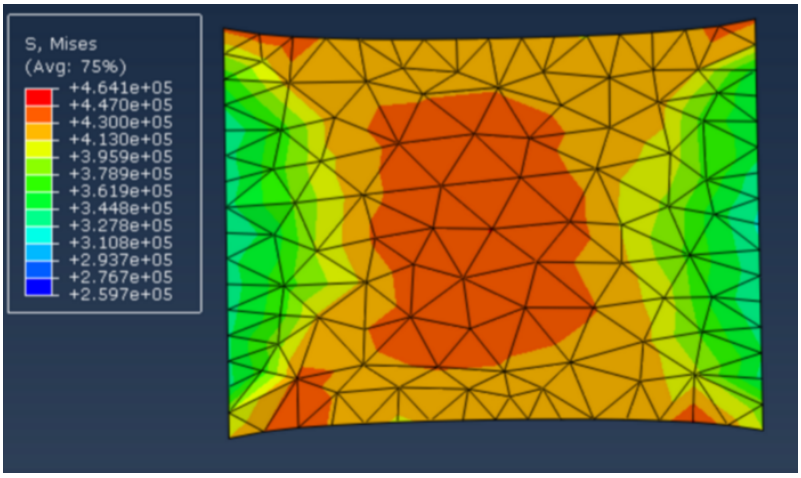
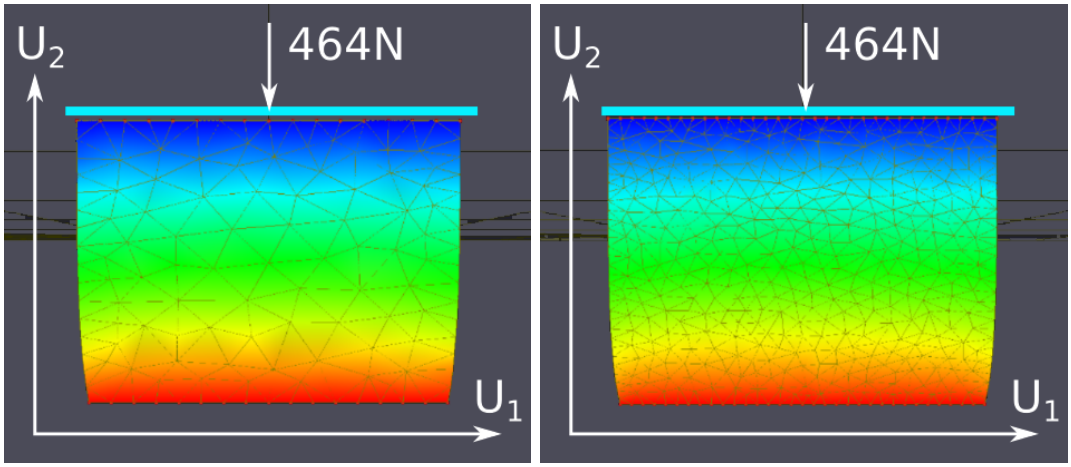


Figure 5.5: The view of the test cube after stretch in Abaqus Explicit. Colour overlay shows the con-Mises stress distribution.



(a) (b)

Figure 5.6: A cube compressed vertically by a weight of 464 N (50kg). The colour indicate the vertical displacement U_2. Two cubes with 244 elements (a) and 21588 elements (b) are shown. The vertical displacement difference is $0.06mm$.

used for time integration depends on the smallest element extent in the mesh. This implies that varying the number of elements (which become smaller when the mesh complexity is increased) in the cube also affects the number of sub-steps, which introduces additional variance that could affect the results of the contact method.

The cube has side lengths of 10cm. The number of tetrahedral elements comprising the cube was changed, while the pressure plate was pushing with the weight of 464 N (50kg). The cube was fixed at the bottom side similar to the previous experiments. Figure 5.6 serves to demonstrate a cube mesh with only 244 elements (a) and a cube 21588 elements (b) subjected to the same compression force. The vertical displacement in both cases is very similar.

The results of the experiments are shown in Table 5.4. They show that the maximum variation in the observed compression of the cube is $0.06mm$, relative to the compression of $24.5mm$ gives an error of 0.25%. This error is relatively small, which indicates consistent results with varying mesh complexity and number of sub-steps. The results indicate that the cube consisting of 244 elements is sufficiently refined to show accurate contact. The difference in the number of required time steps is significant ranging from 61 for the coarser and 487 for the

Table 5.4: Results of varying the number of elements comprising the FE mesh of the cube undergoing vertical compression in BirthView.

# tetrahedra	# sub-steps	compression, mm	error, mm
127	61	24.44	0.06
570	122	24.46	0.04
2887	244	24.48	0.02
21588	487	24.50	0

finest cubes. The results indicate low sensitivity of the contact method to the variation of the time step duration.

Comparison with a gap based method

Cube compression A similar cube as seen in the previous experiment is used in this test. A rectangular block with varying weight is pressed against the top of the cube. The bottom side of the cube is fixed.

The test is run using both a gap based method as described in (Heinstein et al., 2000) and the method based on the pDN method of Yastrebov (2013).

The values of interest are the deformation of the cube and the penetration error that the contact methods allow to persist.

The penalty method allows for a penetration error to persist for reasons described in Section 3.3.5. These errors are dependent on the time step used in the simulation. The TLED update rate is much higher than the rate required to keep the penetration error of the gap based method. However, the BirthView system is running contact detection at a lower rate than the TLED rate for performance reasons. This means that the time-step Δt is larger which leads to larger penetration errors. This can be resolved by decreasing the time-step, but the decreased time-step will increase computation time considerably. The compression magnitudes simulated with a gap based method and the projection based method are reported in Table 5.5

The observed convergence seen in Figure 5.7 could indicate that the projection based method produces the correct results, while allowing for larger time steps. The method appears insensitive to the length of the time-step used for the simulation. The same results are achieved by the gap based approach, but the

Table 5.5: Comparison of the cube compression between the gap and projection based contact methods. The maximum vertical displacement U_2 (compression) is shown for both methods.

	projection based	gap based	
Timestep, ms	U_2 max, mm	U_2 max, mm	difference, mm
0.01	10.61	10.62	0.001
0.1	10.61	10.63	0.002
0.25	10.61	10.73	0.012
1	10.61	11.44	0.083
4	10.61	12.24	1.63
16	10.61	12.37	1.76

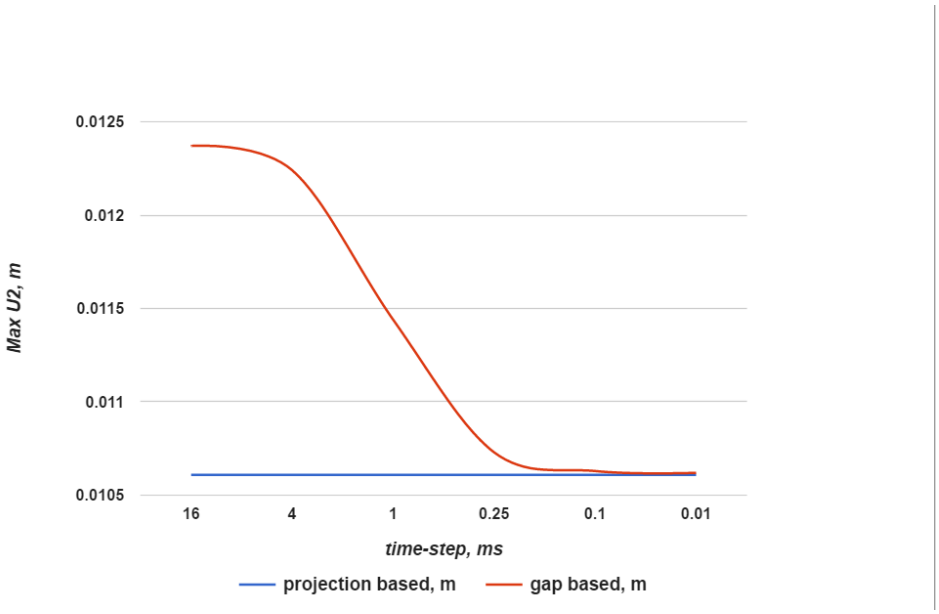


Figure 5.7: The values of cube compression with varying timestep length. The gap based method is seen converging to the values generated by the projection based method as the timestep length is decreased.

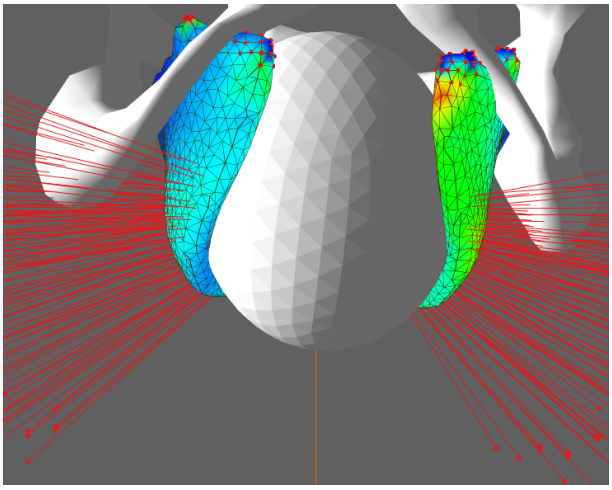


Figure 5.8: An ellipsoid head being propelled through a pelvic floor FE model. Red lines indicate contact normals.

simulation required considerably more computation to achieve them.

Sphere propulsion through pelvic floor Additional experiments comparing the gap and projection based approaches was performed on a more complex setup. An ellipsoid head was propelled through a simple pelvic floor FE model illustrated in Figure 5.8. The vertical descent of the head and the reaction force generated by the contact method are reported. The material properties for the pelvic floor were set to 66kPa and 100kPa for shear and bulk moduli respectively. The time step used for both contact method is amortized to 16ms. Details of amortized contact time steps is described in Section 4.2.13. The vertical displacement and the contact reaction force are reported for both the gap and projection based methods.

Figure 5.9 demonstrates the discrepancies in the reaction force. As seen previously the gap based method produced lower reaction force values when the time-step was larger. This results in the head descending faster as seen in Figure 5.10. Reducing the time step by a factor 10 resulted in a more similar behaviour to the projection based contact.

When the time step was reduced 10 times down to 1.6ms the simulated results became more similar as seen in Figures 5.11 and 5.12. The behaviour of the head is closely similar as illustrated by the error curve, which only peaks near to

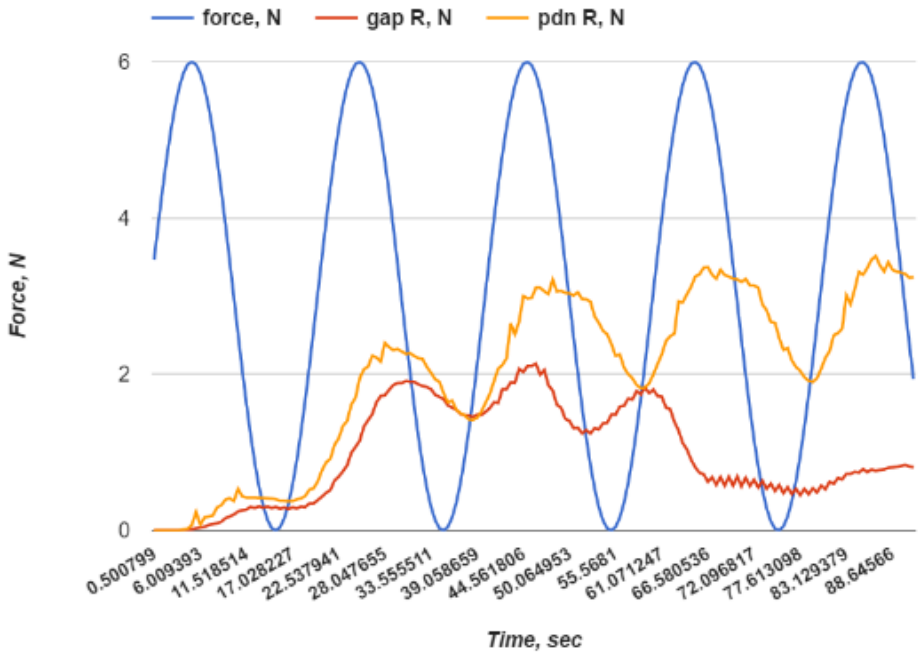


Figure 5.9: Reaction force for the gap (yellow curve) and projection (red curve) contact methods during simulated ellipsoid head propulsion. The blue curve shows the expulsion force acting on the head. The time step used for this simulation is 16ms.

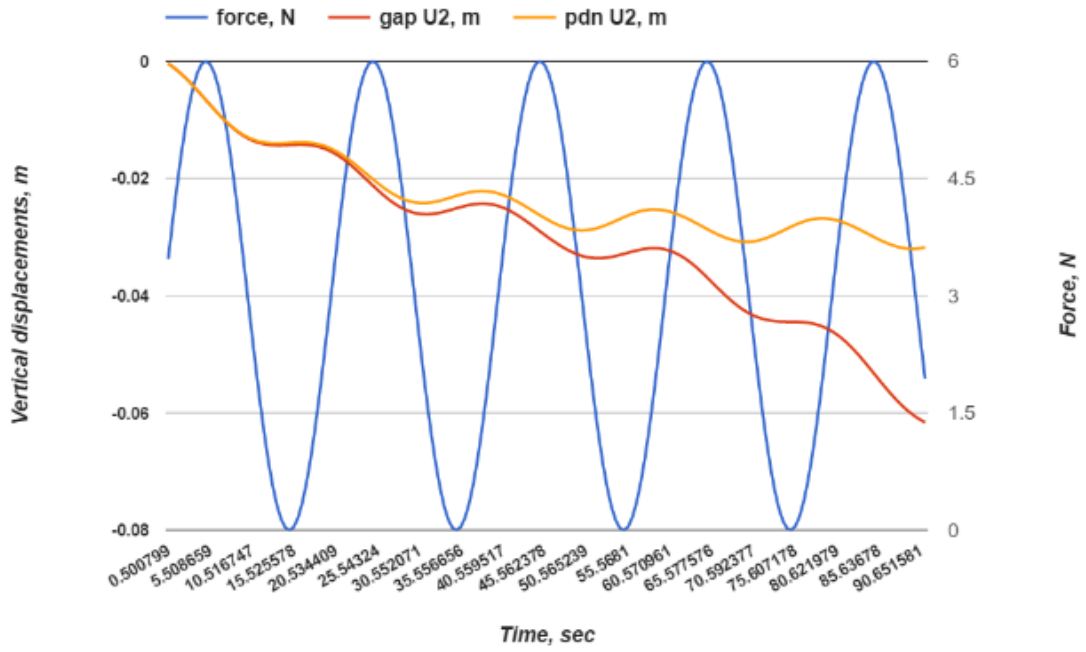


Figure 5.10: Head's vertical position for the gap (yellow curve) and projection (red curve) contact methods during simulated ellipsoid head propulsion. The blue curve shows the expulsion force acting on the head. The time step used for this simulation is 16ms.

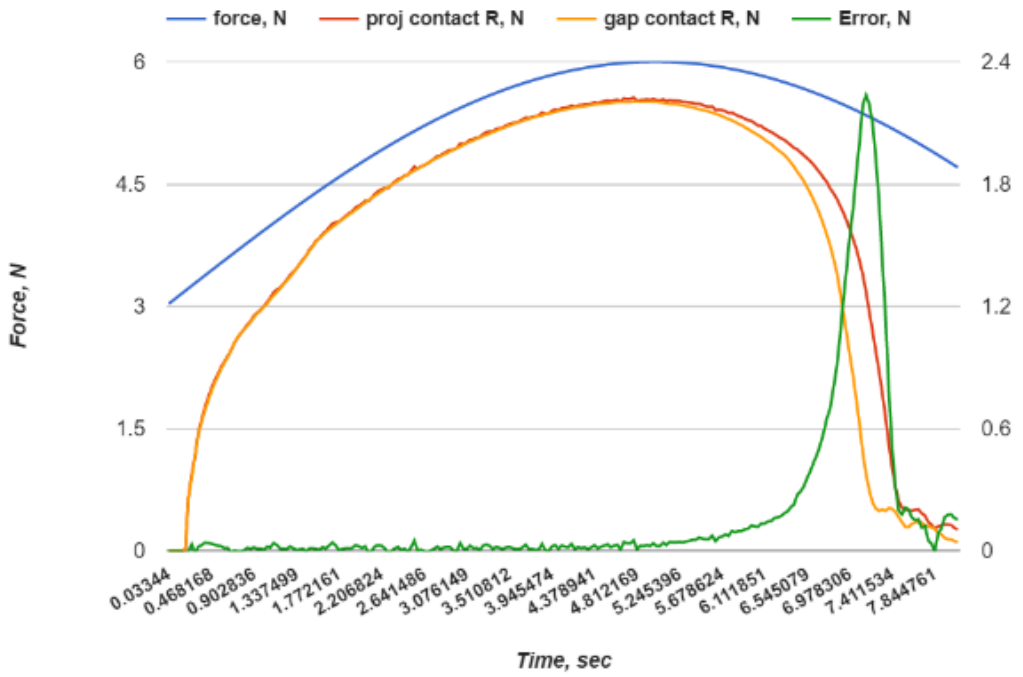


Figure 5.11: The reaction force for the gap (yellow curve) and projection (red curve) contact methods during simulated ellipsoid head propulsion. The blue curve shows the expulsion force acting on the head. The time step used for this simulation is 1.6ms. Note that the time range shown in this figure is different from Figure 5.10

the end of the simulation as the errors accumulate. The curves for the vertical displacement only diverge at the end.

Note that the graph in Figure 5.9 is demonstrating the much shorter time period when compared to the one in Figure 5.11. This is due to the fact that the reduction of the time step with a factor 10 requires increasing the amount of computation performed per frame 10 times. Increasing the amount of computation results in a super-linearly increased simulation time, i.e. where the simulation with 16ms time step took approximately five minutes, the 1.6 ms simulation took more than two hours.

The contact in the latter figure is initiated manually by moving the fetal head towards the pelvic floor to speed up the process. The difference in the reaction force magnitudes are also caused by the slightly different trajectory taken by the head as compared to the former experiment due to the difference in the initial

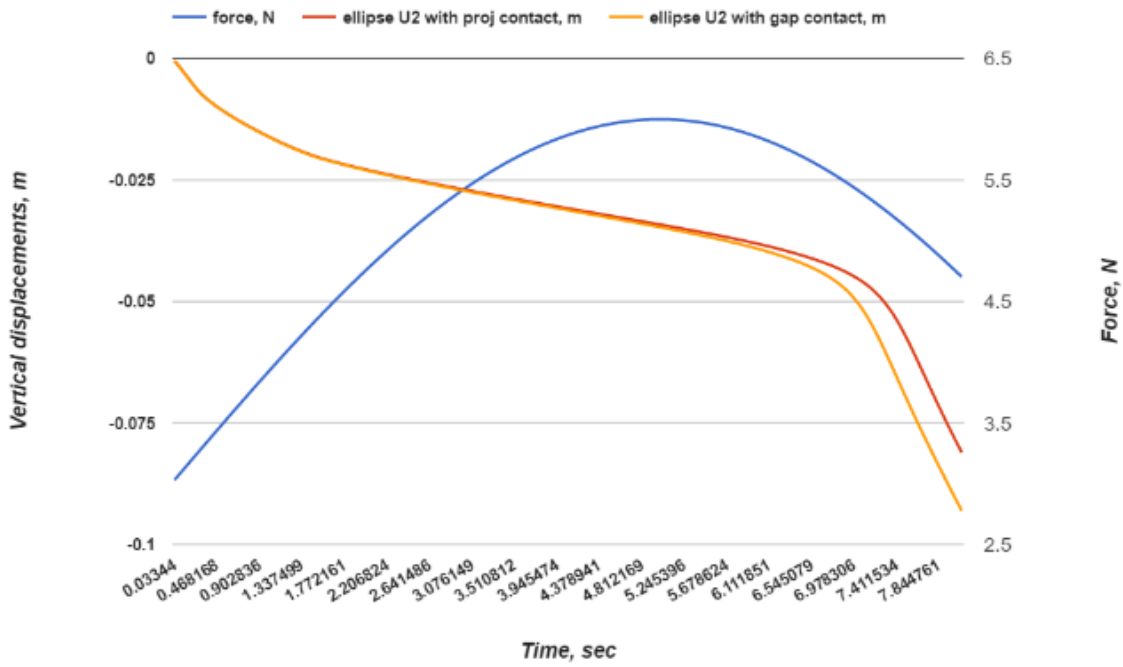


Figure 5.12: The head's vertical position for the gap (yellow curve) and projection (red curve) contact methods during simulated ellipsoid head propulsion. The blue curve shows the expulsion force acting on the head. The timestep used for this simulation is 1.6ms.

contact. The main observation is that the reaction forces from the projection and gap contact methods converge with smaller time steps.

5.2.3 Assessing GPU TLED performance

CPU vs GPU performance

The actual speed-up values can be observed, when the OpenCL kernels are executed on the CPU as normal programs. The performance results of executing these on the CPU can be compared with the results of GPU execution.

The OpenCL kernel is compatible with C language and with small modifications they can be compiled as part of the application executed on the CPU.

A series of experiments were performed to identify how the GPU implementation performs when compared to the CPU implementation. Figure 5.13 illustrates how the number of number of iterations per second (IPS) varies with the number of elements in the FE mesh. Note that for smaller numbers of elements the CPU implementation shows better results, as the GPU processing has overheads.

The number of IPS only slowly goes down for the GPU with the increase of the number of elements. All the operations performed on the CPU and the data transfer introduces large overheads leading to lower simulation rates than the number of IPS would suggest. In fact the 1.8 million elements cube took 15 milliseconds per sub-step on average when considering all the GPU overheads. The number of sub-steps required for convergence due to the length of the time step is 195 per 16 milliseconds. This leads to 2.5 seconds of computational time per 16 milliseconds of simulated time. This implies it is 156 times slower than the required real-time simulation rate of 60 updates per second. Almost near real-time simulation rates were only achieved at around 100k elements.

Selective mass scaling and small element removal

Selective mass scaling is an approach used to improve the critical time step (Olovsson et al., 2005). Introducing mass scaling to the assembly can greatly reduce computational costs of the explicit simulation without degrading the accuracy.

The main principle of general mass scaling is the increase of density of the simulated tissues. Increasing the overall density of the object, however, can lead

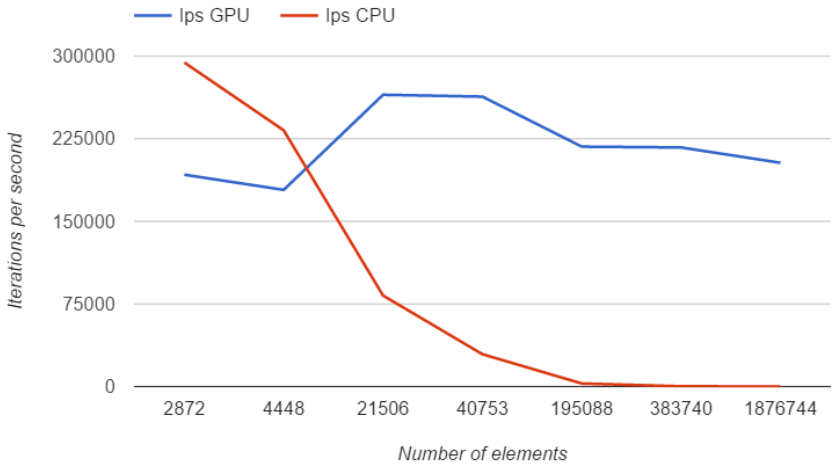


Figure 5.13: Performance comparison between the CPU and GPU TLED implementations. Iterations per second (ips) for CPU and GPU against the number of tetrahedra in the cube.

to unrealistic simulation results as the object behaves as if it is heavier than it should be. Selective mass scaling implies only increasing the density of the elements that are responsible for the decreased critical time-step.

Olovsson et al. (2005) present precise methods of applying selective mass scaling depending eigen-analysis of the stiffness matrix. We on the other hand apply a more simple technique. The process begins with identifying the elements with highest stiffness and the shortest minimal extent. These are the elements of which the density is increased 10 times. We find that these elements are relatively sparse and are small, which means that the effect of increased mass will be minor on the overall behaviour of the object.

Recall Equation 3.45 that features the material density in the denominator. This means that larger density values will decrease the speed of sound. From Equation 3.44 we see that reducing speed of sound will increase the critical time step.

5.3 Experimental setup

The main task of the experimental setup in the case of a virtual simulation is to generate a virtual scene which recreates a real phenomenon with as much accuracy as possible. The objects taking part in the real phenomenon must be represented by one or more virtual objects. The behaviours of these virtual objects must approximate the ones of the real objects as well. Furthermore, the interactions between the objects must be replicated such that the results of said interactions are the same or approximately as the real ones.

In the case of surgical simulations FE model needs to predict the deformation *field* within a particular organ that the surgery is aimed at. The *boundary conditions* are the attachment tissues that connect and fix the organ to the body. The *external loads* are the forces and deformations that are applied from the virtual surgical tools when they are in contact with the organ.

The fetal skull is the most important fetal organ and should be represented in the experimental setup with much accuracy. The fetal shoulders could be of particular importance when considering the shoulder dystocia cases, but the purposes of studying the cardinal movements the shoulder are assumed to be of little consequence.

5.3.1 Fetal model

The model of the simulated fetus consists of two separate parts. The fetal skull is one of the main objects in the simulation and is represented by a rigid body. The description of the fetal head model is provided in Section 3.1.3. The second part is the fetal trunk representing a simplistic model approximating the shape of the fetal trunk.

The fetal head is attached to the fetal trunk, but all rotation movements are not inhibited by the attachment. A basic spring based attachment system is used for the simulation of the neck.

Apart from the translational effects, the neck spring is also capable of exerting torque on the attached objects.

For the purpose of the simulation, the tolerance angle of 30 degrees is used to emulate the free rotation of the head. Any rotation further than the tolerance

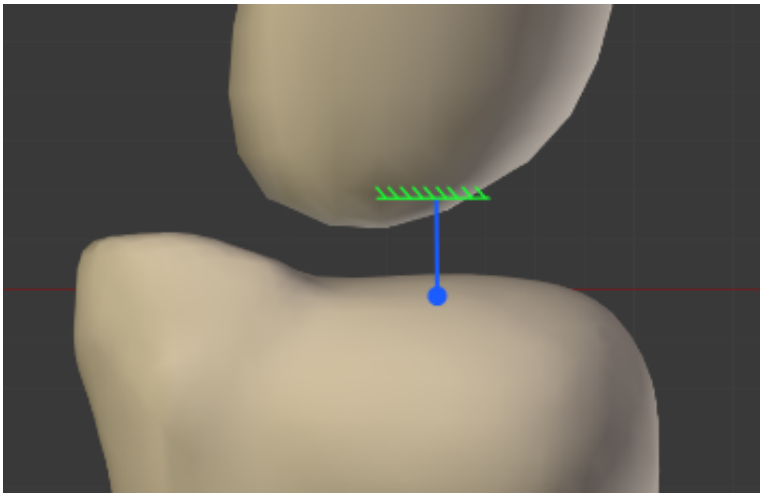


Figure 5.14: The setup used to simulate the basic fetal neck model. The neck is modelled as a deformable rod using a linear bending spring.

threshold causes a resistive torque to be exerted on both the attached objects. The torsional stiffness used for the simulation is $100\frac{N \cdot m}{rad}$.

5.3.2 Material properties

Rigid objects

The head of the fetus is chosen to be rigid. The model of the fetal head is based on the one presented by Lapeer and Prager (2001). The original mesh of the skull has been preprocessed by decimating and smoothing as described in Section 3.1.3.

The maternal pelvis is also set to be rigid. The two main regions of the maternal pelvis are the pubic symphysis and the coccyx. The deformations occurring in the region of the symphysis pubis are minor, when compared to the deformations of the soft tissues, and is limited to the widening of the birth canal(Gabbe et al., 1991).

One of the main reasons for the preference of rigid structures for simulation objects is dictated by the nature of TLED FEA. The deformable models for pelvic bones should have material properties with high elastic moduli, in the order of 100MPa (Dalstra et al., 1993). Such large elastic moduli would make

the simulation highly demanding and slow down the simulation rate down out of real-time range. Similar issues are present when simulating the deformable fetal head. The potential for a simulation including full fetal head moulding is discussed in Section 6.3.

Deformable objects

The choice of material properties used for the primary simulation of the pelvic floor is based on the values reported by Hoyte et al. (2008). The values for a Neo-Hookean hyper-elastic material of the pelvic floor is assigned are $66kPa$ for the shear modulus and $1MPa$ for the bulk modulus. The value of $60kPa$ used by Lepage et al. (2015) for the pelvic floor muscles also lies closely to the value used in our childbirth simulation. Note that the value for the bulk modulus is chosen lower than the nearly incompressible value of $1GPa$ used by Hoyte et al. (2008). The overly large values for the modulus cause instabilities in the simulation that the compressible Neo-Hookean material response model.

The damping coefficient c was chosen be 3500 for all soft tissues in the simulation. The high value for damping was beneficial for the stability of the simulation due to it reducing the kinetic energy. Highly dynamic processes resulted in unstable simulation and a high damping value prevented fast motions from occurring. Such a high value was established empirically to achieve a visually plausible motion of the soft tissues. The highly constrained situation of the soft tissues of the mother and the high liquid content in the pelvic cavity are assumed to justify using a high damping value.

Summary

A collection of the information about the materials used in the simulation is provided in Table 5.6.

Fetal head initial position and orientation

For the simulation of normal labour the fetal head was positioned just above the pelvic brim. The distance between the head and the brim was chosen approximately without particular quantitative basis. The head was positioned above

Table 5.6: The types of objects present in the simulation and the material properties used to simulate them.

	material type	material properties
pelvis	rigid	
trunk	rigid	
head	rigid	
neck	linear spring	$100\frac{N}{m}$
	torsion spring	$100\frac{N \cdot m}{rad}$
pelvic floor	Neo-Hookean	μ 66kPa and K = 1MPa
ligaments	Neo-Hookean	μ 160kPa and K = 1MPa

the pelvis instead of pre-engaged into the pelvic inlet so that the simulation can distinctly demonstrate non-imposed descent and engagement.

The orientation of the head was chosen to be facing transversely. This initial orientation is chosen based on (Gabbe et al., 1991). Initial flexion is not imposed on the head.

5.3.3 Periodic expulsion force

Magnitude

The uterine expulsion force in an average case of the second stage of labour for a Caucasian woman (Riethmuller and Schaal, 2012) can be described by a periodical function. A sinusoidal function was chosen as the basic shape of the periodically changing expulsion force.

According to Misra (2006) the intra-uterine pressure during the first stage of labour is normally at 20-30 mmHg. During the second stage of labour the values 50-60 mmHg. It is unclear whether or not the growth is gradual, but we chose to model the force as gradually increasing. The magnitude of the expulsion force is modelled as a slowly growing value as seen in Figure 5.15. The frequency of the contractions also increases during the second stage of labour. The change in the frequency is not modelled in our simulation system, but the more frequent

contractions can be roughly approximated with the increased expulsion force.

The magnitude of the expulsion force is calculated based on values reported by Ashton-Miller and DeLancey (2009). The magnitudes of 16 N at rest, 54 N during a uterine contraction, and 120 N during a volitional push are used in our simulation. The value of 120 N was found to be insufficient for the delivery of a fetus with larger head, so the maximum expulsion force was increased to 200 N. That is, the expulsion force will grow until the head is delivered or the value of 200 N is reached.

We did not separate the distinct effects of uterine expulsion and the maternal volitional push into separate phenomena and apply the combined full expulsion force on the fetal bottom. The net effect from uterine pressure will be towards birth canal allowing the combination. However, the uniform application of the pressure onto the full fetal head could be more realistic and should be investigated in future works.

The minimum expulsion force is imposed on the simulation. The justification being that there is always a pressure acting on the fetus pushing it downwards. It is referred to as the "base tonus" pressure by Buttin et al. (2009).

There is a period of relaxation between contractions in real birth (Buttin et al., 2009). The lack of relaxation in our model is based on the fact that the time dependent effects of the period would only be relevant if visco-elastic material properties were used for the pelvic floor muscles.

Application point and direction

The point of the expulsion force application is chosen to be at the fetal buttocks. As suggested by Buttin et al. (2009) the expulsion force has two components, i.e. the involuntary uterine expulsion force and the voluntary force from the abdominal muscles of the mother. The uterine pressure should be applied uniformly on the surface of the fetus, whereas the abdominal muscle expulsion acts mostly downwards on the bottom of the fetus. This model is used by Buttin et al. (2009). In our case a single force pushing on the fetal buttocks along its spine is used.

A frame of reference should be introduced when discussing the force directions. Figure 5.16 shows the pelvic inlet (green circle) and a perpendicular axis (green

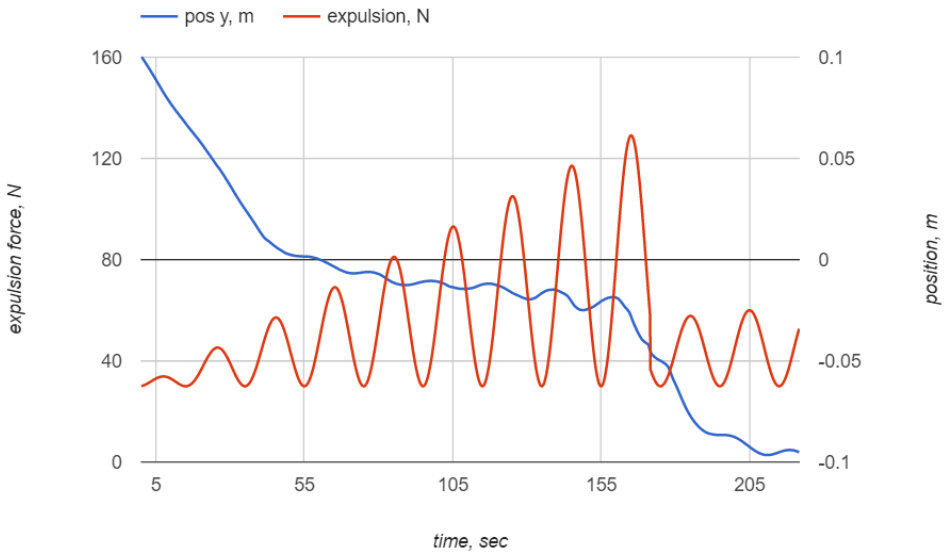


Figure 5.15: The plot of the increasing periodic expulsion force. The vertical position of the fetal head (station) is shown by the blue curve. The expulsion force is reduced after the delivery of the head.

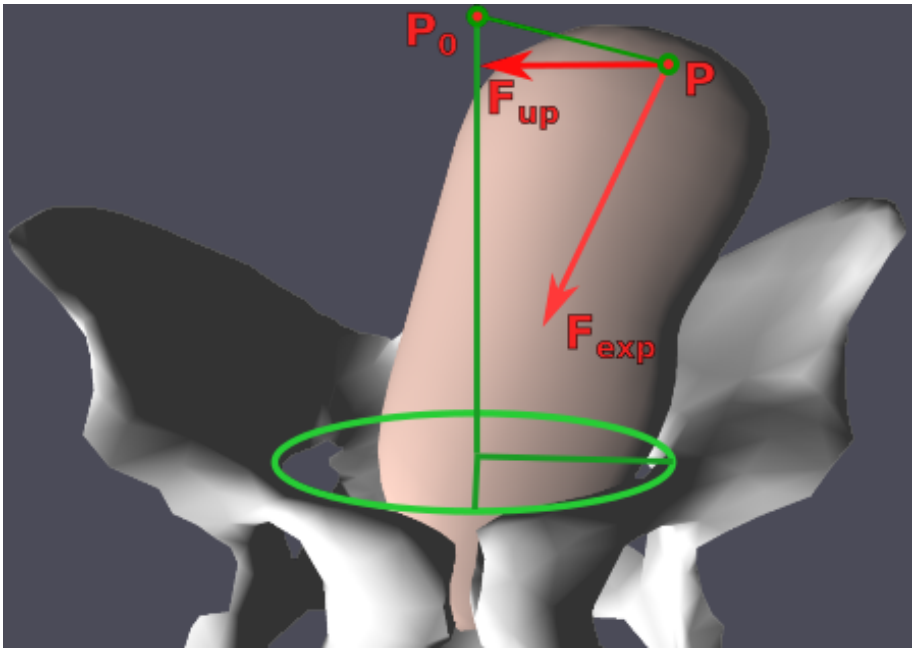


Figure 5.16: The fetal trunk after the head descended into the pelvic cavity. The explusion force F_{exp} has application point P. The force F_{up} is pulling the trunk in the upwards position, while not introducing any pull along the trajectory of the fetus. The green circle marks the pelvic inlet. The green line perpendicular to the circle marks the "vertical" axis.

line) pointing away from the pelvis. In the context of this childbirth simulation, this axis is referred to as the vertical axis. Note that, in real childbirth the axis will depend on the orientation of the mother. When this text refers to the vertical direction, the direction along the axis perpendicular to the pelvic inlet is implied.

The actual direction of the force is determined by the orientation of the fetal trunk. That is the force is always acting along the central axis of the fetal trunk along the fetal spine. The force is thought to propagate through the fetal body and would effectively be applied in the direction of the central axis. Arguably, the direction of the force could be strictly in this direction at all times, considering that the expulsion force is partially transferred through the amniotic fluid and will act on the fetal head directly pushing it downwards. However, the trunk is kept vertical (along inlet perpendicular axis) by a spring, thus effectively causing the expulsion force to be almost always vertical. The trunk is allowed to undertake minor direction changes as the linear spring will allow some motion when a larger toppling torque is acting on the trunk.

The vertical orientation is essential to the success of simulating childbirth. The orientation could be simply prescribed by forcing all the rotations of the trunk to be cancelled. This would result in the trunk only being able to rotate around the centralaxis. However, this behaviour is seen as unrealistic as the trunk is suspended in the amniotic fluid and can undertake slight rotations around non-vertical axes as well.

The force preserving the downwards direction of the fetal trunk is calculated using a linear spring. Figure 5.16 is showing how the fetal trunk is kept upwards. The vertical orientation of the trunk is maintained through the use of a spring attached to the bottom of the trunk, which is where the expulsion force is applied as well.

Note that simply connecting the spring to the original point P_0 will result in the introduction of spurious vertical forces that will affect the expulsion force, as the force has a component along the vertical axis. The direction of the force is kept strictly vertical, by projecting the force onto the horizontal plane. This is demonstrated in Figure 5.16.

This approach is an approximation of the behaviour of the fetal trunk when enclosed in the maternal uterus. Once the fetal head engages into the maternal

pelvis, the uterus surrounding the trunk will act to prevent the toppling of the fetal trunk, thus maintaining the trunk direction mostly vertical. The basic linear spring attached to the bottom of the fetal trunk serves to approximate the uterine influence.

5.4 Experiments

5.4.1 No soft tissues

The initial experiments of the childbirth simulator were dedicated to determine if the presence of the rigid maternal structures and the bony fetal skull would be sufficient to observe simulated cardinal movements. The contact model was also a basic model applicable only to contact between rigid bodies. The work presented in paper by Gerikhanov et al. (2013) describes the findings of these initial experiments. The results indicate that the full set of cardinal movements cannot be simulated with only the rigid maternal bony structures and rigid skull present in the simulation.

As reported in the paper, the occurrence of engagement and flexion was consistent as long as the fetal head had a sufficiently large radius to engage with the parts of the pelvic inlet.

To further investigate if the presence of the ischial spines is sufficient for the the occurrence of the internal rotation, the fetus was rotated to the ROA presentation. With the ROA presentation none of the cardinal movements were observed for the simulation with no FE pelvic floor. These results are supported by the assertions made by Veit (1581) and T'arnier(1888) in their papers where they showed that the presence of the bony pelvis alone is insufficient for internal rotation to occur.

For the sake of completeness, we do cover the experiments with only the rigid bony structures of the fetus and the mother in Section 5.5.

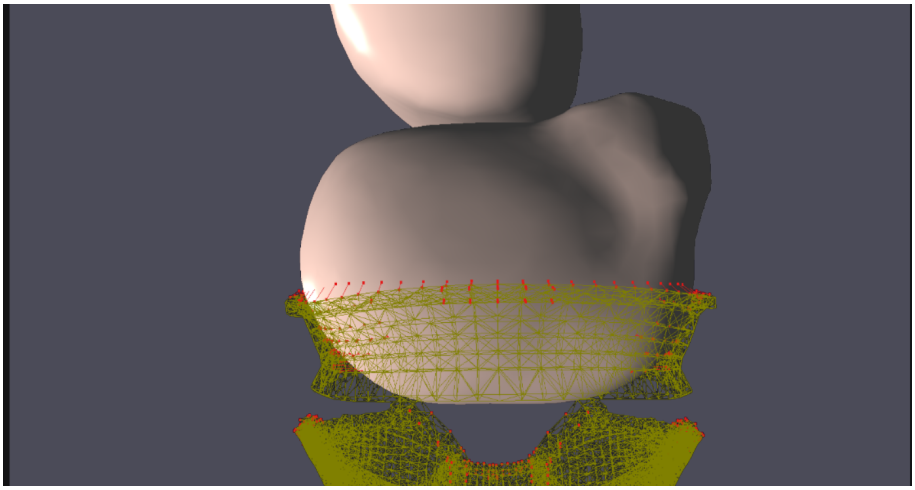


Figure 5.17: The head engaging the cervix. The initial orientation is transverse and immediately above the cervix. The trunk is located above the head at 2cm.

5.4.2 Normal birth simulation

The main set of experiments done for the purposes of this thesis are concerned with the simulation of normal childbirth. The detailed description of what is considered as normal birth is given in Sec. 1.3 and is generally considered to be fulfilled in the cases when the baby is vaginally delivered without harm and the cardinal movements were observed. The main aim of the experiment is to observe the occurrence of the cardinal movements in the simulated scenario. Over time, as the development of the simulation was iteratively updated a large number of modifications were required prior to the cardinal movements being observed. The vast majority of these were minor modifications to the setup and tweaking and are not reported here. Only the essential few that had considerable effects on the results of the simulation are reported.

Flexion

As mentioned before, the initial position of the fetal head is set to be slightly above the pelvic inlet. This allows the simulation to manifest the initial stages of the descent and the engagement. The orientation of the head is transverse and no initial flexion or extension is imposed. This initial setup is shown on Fig. 5.17.

In order to quantitatively evaluate the magnitude of the observed flexion, the

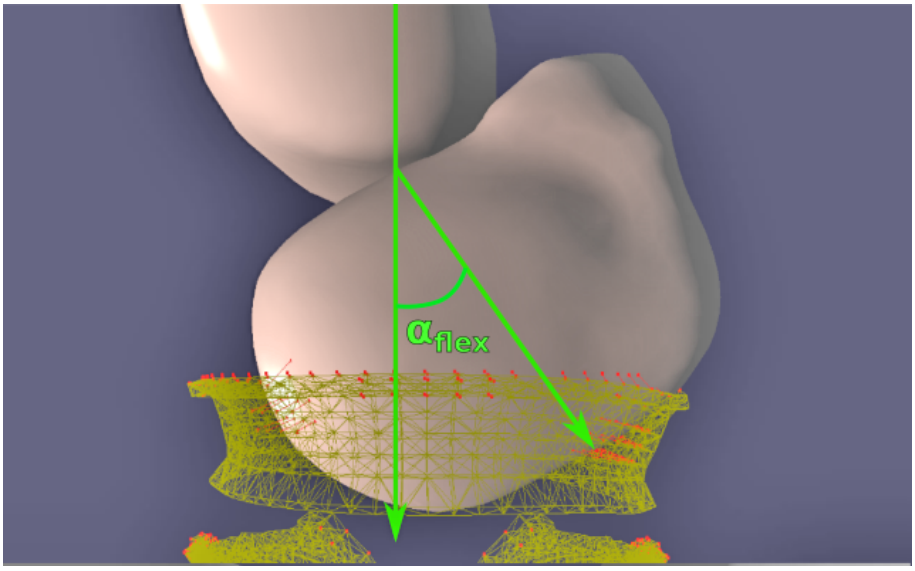


Figure 5.18: Angle between vertical axis of the head and the maternal central axis.

angle between the vertical axis of the head and the vertical axis of the trunk. In fact the flexion is measured as the angle of the head's rotation around the transverse axis passing through the shoulders of the fetal trunk. This angle is demonstrated in Figure 5.18.

Internal rotation

The internal rotation is one of the most important cardinal movements because the fetal head may - and often will - be arrested in the narrow transverse diameter of the pelvic outlet if it does not undertake the internal rotation.

Qualitatively speaking the internal rotation is the most critical to simulate. If the head fails to do the internal rotation none of the further cardinal movements will be observed in the traditional sense. Extension and expulsion are possible, but will occur inconsistently and most often will lead to arrests and/or unrealistic deformations of the pelvic floor.

We have shown that the presence of the maternal bony pelvis is insufficient for the consistent manifestation of the internal rotation (Gerikhanov et al., 2013). In the paper, the internal rotation was observed partially.

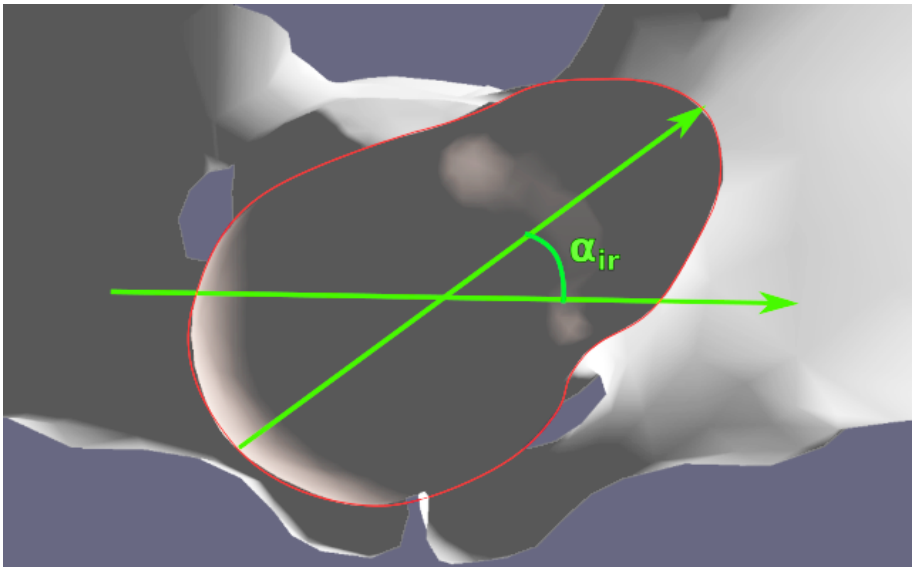


Figure 5.19: The angle between the longitudinal axis of the head and the maternal transverse axis is used to measure the internal rotation.

In the current experiments, the magnitude of the observed internal rotation is measured through the angle between the longitudinal axis of the fetal head and the maternal transverse axis. The angle is demonstrated in Figure 5.19.

Additionally, the forces and torques causing the internal rotation can be studied using the simulation system. The ones contributing to internal rotation and their sources should be identified.

Extension

The measurements of the extension are obtained in the same fashion as the measurements of the flexion. The extension, however, will be observed as negative flexion. That is, when the head is flexed by 45 degrees, its extension of the same magnitude will be reported as -45 degrees flexion.

Rotation of the fetal shoulders

The initial position and orientation of the fetal trunk can be seen in Figure 5.17. The trunk is above the fetal head with a clearing of around 2cm approximating the fetal neck.

The main task of this experiment is to investigate the rotation of the fetal shoulders. Considering that the trunk is a rigid object, the rotations of the shoulders can only occur with the rotations of the whole trunk.

Borell and Fernstrom (1958) define two distinct events of rotation by the fetal shoulders. The first event occurs in the area above the pelvic inlet and is typically seen as a rotation from the sagittal/sideways orientation to the coronal/frontal orientation. The second rotation is undertaken by the shoulders in the area of the pelvic outlet and is rotated back to the sagittal orientation in the majority of cases (Borell and Fernstrom, 1958).

5.5 Results

5.5.1 Simulated flexion

The process of flexion can manifest itself at the same time as the initial cardinal movements are performed. Which implies it being a continuous process overlapping several other processes. In the results of our simulation, however, the majority of the flexion occurs after the head comes into contact with the upper portions of the pelvic floor.

Mechanics

The fact that both the pelvic floor and the bony pelvis are crucial for flexion to occur is apparent from the simulated results as full flexion is not observed consistently when either one is absent. The presence of the bony pelvis is essential, but it is the protruding ischial spines that initiate the flexion. The pelvic floor in turn is the main cause for the occurrence of further flexion. The visualization of the contact forces indicates that the regions of the head in contact with ischial spines are the ones mainly contributing to the torque required for the flexion to occur. Fig. 5.20 shows the areas of contact between the head and the ischial spine region of the maternal pelvis and the contact with the pelvic floor respectively during flexion.

The contact with the ischial spines produces two contact forces that are asymmetric. The asymmetry causes a torque on the head and forces it to flex. After

the internal rotation occurs, the head is no longer in contact with the ischial spines, but the flexion is maintained by the contact with the sacral and coccygeal area of the maternal pelvis.

Forces acting against over-flexion

The fetal head has constant flexing torque acting on it. This torque would cause the head to over-flex whereby the occipito-frontal axis is fully vertical. This is prevented due to the interaction between the fetal mandible and the chest of the fetal trunk. The contact preventing further flexion is shown in Figure 5.21.

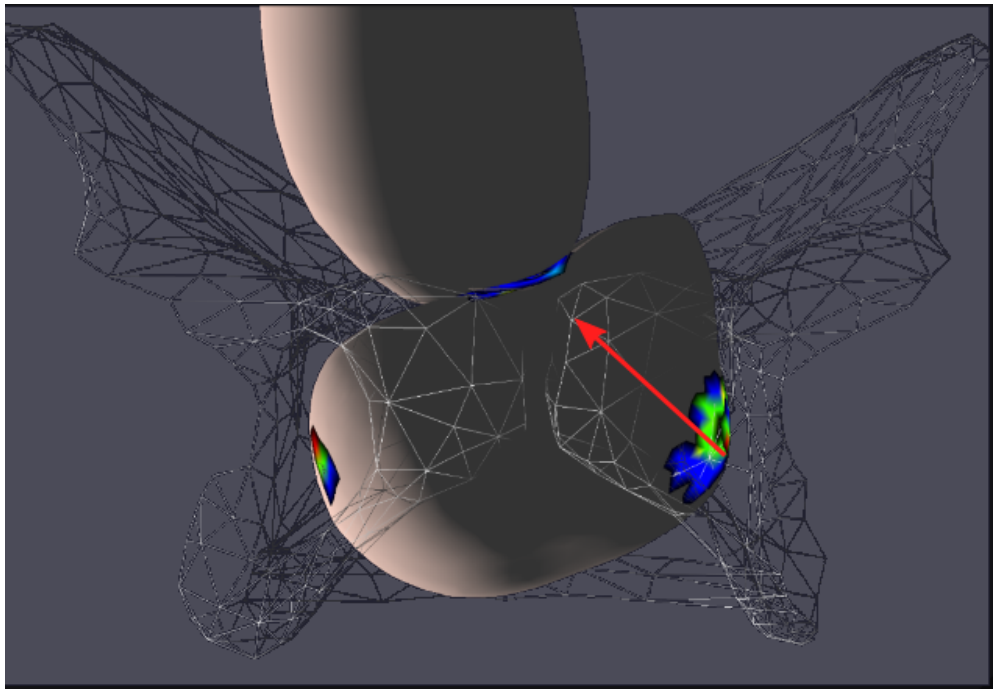
5.5.2 Simulated internal rotation

The internal rotation was successfully simulated by the BirthView simulation system after a few modification. A more consistent simulation required additional parameters to be incorporated into the model. The internal rotation was partially observed even with only a bony pelvis and no soft tissues but only after adding soft tissues did the internal rotation occur fully and consistently. The presence of the sacrospinous ligaments was found to be of major importance when observing the internal rotation and consequently to the outcome of labour. The ligaments seem to play an important role in supporting and guiding the rotation of the fetal head while it undergoes the internal rotation. The pelvic floor is also important as it functions a pivoting support for the head while it rotates.

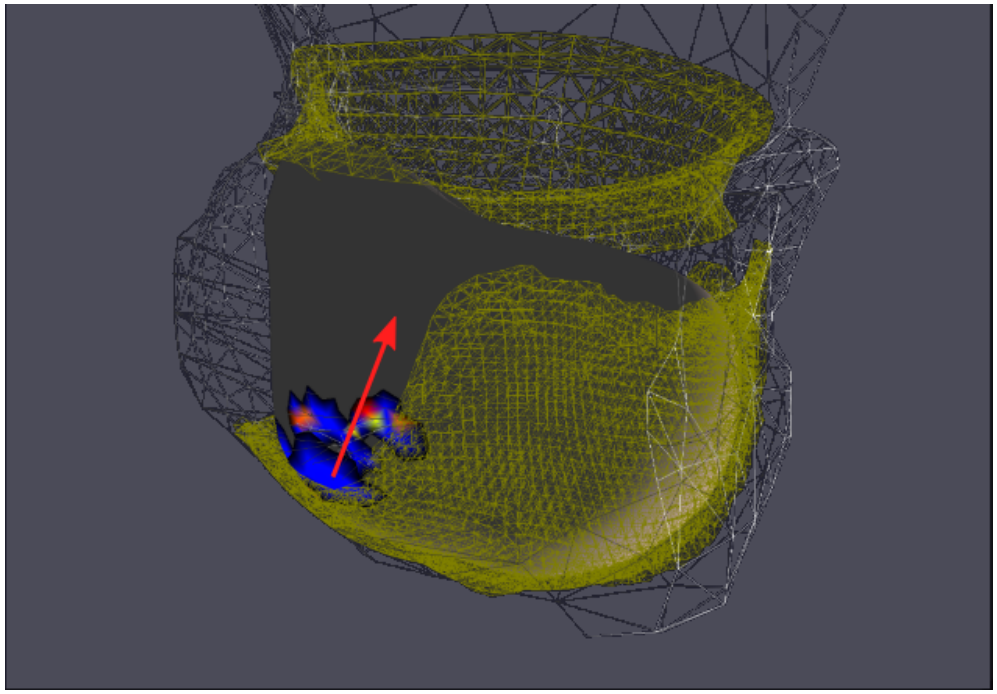
Bony pelvis only

Our previous experiments from (Gerikhanov et al., 2013) have been repeated. The experiments show no full internal rotation after engagement and descent with only the bony pelvis and the fetal head present. Some internal rotation can be observed due to the presence of the ischial spines, but the head fails to rotate fully as it descends further into the pelvic cavity. Ultimately the head exits through the pelvic outlet at an oblique angle. This can be seen on Figure 5.22. It should also be noted that full flexion did not occur.

When the simulation features only the bony pelvis, most of the cardinal movements are not observed which is in line with the findings by Varnier (1900).



(a)



(b)

Figure 5.20: The locations of contact between the fetal head and the pelvis (shown in wireframe) leading to the occurrence of the flexion. Flexion prior to the internal rotation due to contact with ischial spines (pelvic floor is not shown to avoid occlusion) (a). Side view of pelvis after internal rotation, where the flexion by contact with coccyx area and the pelvic floor (shown in wireframe) (b).

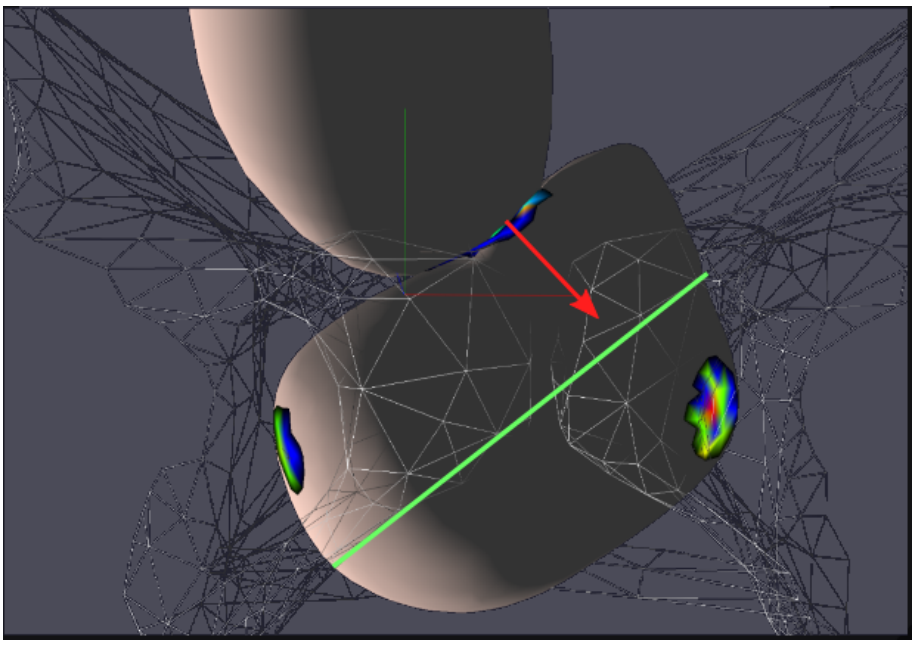


Figure 5.21: Contact with trunk stopping the head from flexing further. Arrow indicates the direction of the contact force acting against flexing torque. The occipito-frontal axis is shown as a green line.

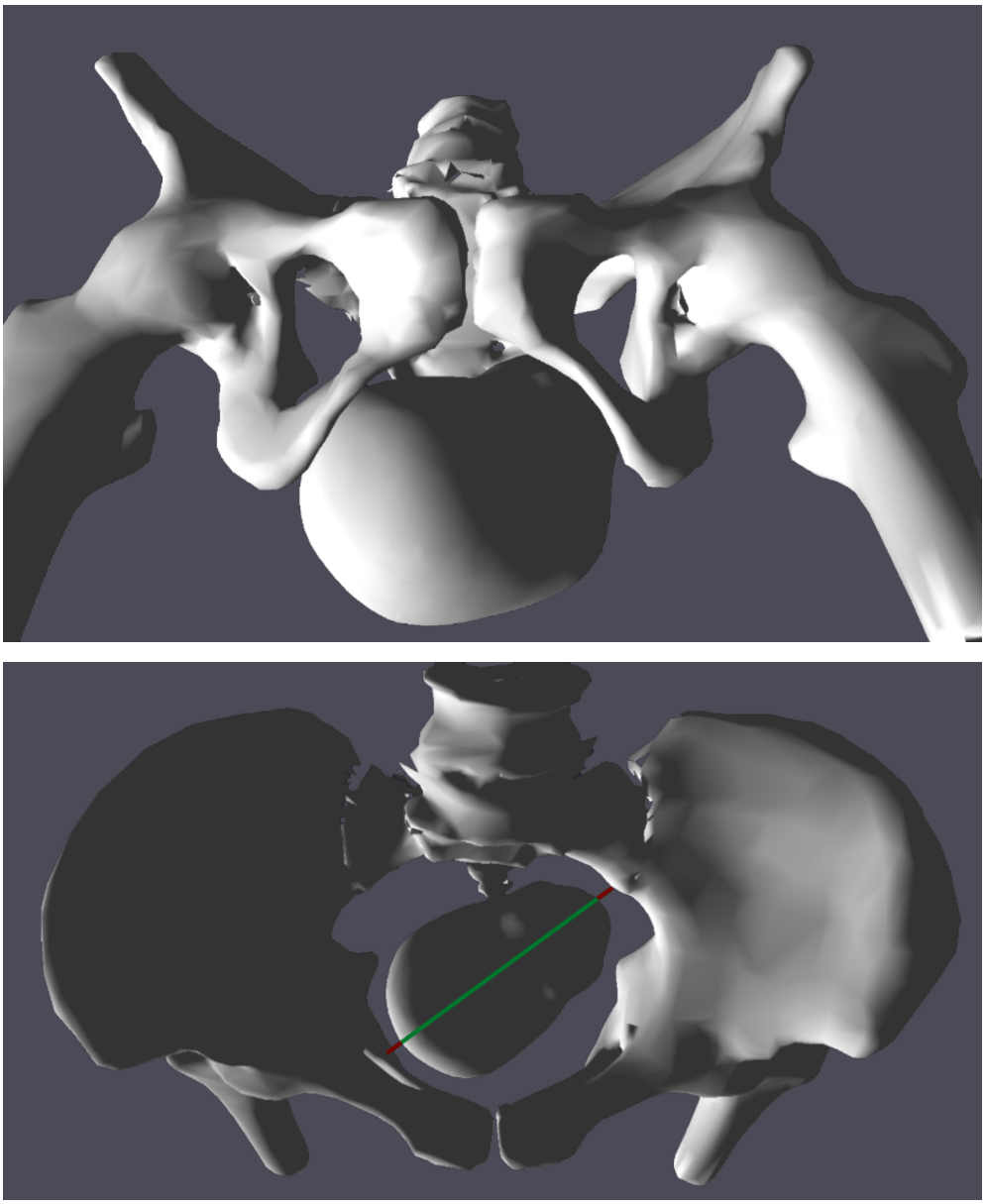


Figure 5.22: Fetal head immediately after it negotiates the birth canal with only the bony pelvis present. The frontal view of the pelvis (top) being born. Top view of the pelvis (right) shows the head failing to fully internally rotate, staying at approximately 45 deg.

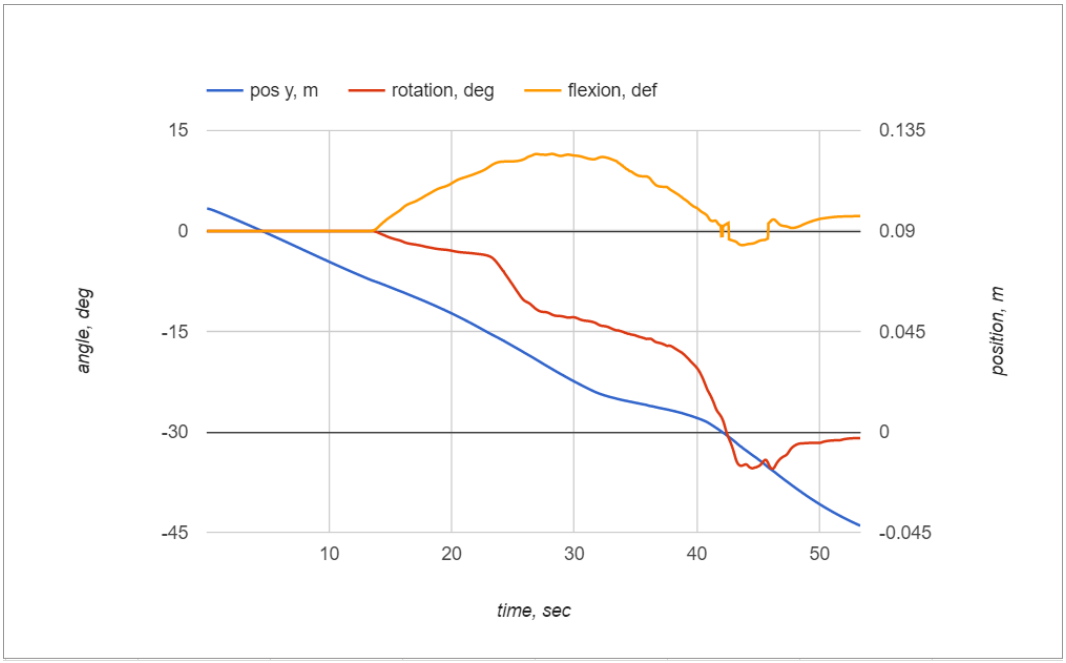


Figure 5.23: Graph of fetal head motion with bony pelvis only.

Varnier (1900) undertook an experiment where the maternal pelvis was eviscerated of all soft tissue and only bones were left and the cardinal movements were not observed. The rotation and flexion of the head are plotted against time as seen in Figure 5.23.

The head can be seen extending throughout the simulation, but later returns to its un-flexed orientation. The rotation can be seen growing until reaching around 35 degrees and then oscillating slightly.

The vertical (longitudinal axis of the birth canal) position of the head is also presented. It can be seen that the movement of the head is virtually unimpeded and progresses fast. As expected this leads to shorter than normal duration of childbirth.

Bony pelvis, pelvic floor and cervix

When a pelvic floor model is incorporated into the bony pelvis the head behaves differently as compared to the case where only the bony pelvis is present.

It is important to mention that further descent of the head is prevented by the

pelvic floor and the fetal head becomes arrested in the pelvic cavity. Additionally, the head undergoes full flexion when it makes contact with the pelvic floor. Figure 5.24 illustrates the behaviour of the head while its exiting the birth canal. The internal rotation did not occur fully, but rather the head would engage into the greater sciatic foramen and stay at approximately 45 degrees of internal rotation (red curve in figure stops from descending further).

The graph illustrating the motions of the fetal head is shown in Figure 5.25. As can be seen from the graph, the rotation is incomplete and stops after reaching the value of approximately 45 degrees. Flexion can be observed, but no extension is manifested at the end of the graph. Importantly, the position of the head starts to oscillate at a constant value, indicating the head fails to progress further as it is arrested by the pelvic floor.

Bony pelvis and sacrospinous ligaments

Introducing the sacrospinous ligaments to the model did not yield any additional cardinal movements after the descent and engagement. No noticeable difference in the motion of the skull was observed as compared to the simulation with only the bony pelvis present, except that the contact between the ligaments and skull prevented the skull from entering into the greater sciatic foramen.

Figure 5.26 shows what the final orientation of the head is after 30 seconds. It can be seen that the orientation of the head is oblique at approximately 45° to the transverse axis of the pelvis.

While descending, the head comes into contact with the left sacrospinous ligament. The contact leads to the head being pushed anteriorly, so that it does not enter the foramen.

A bony pelvis with hard ligaments leads to the arrest of the fetal head. This only occurs when the head is in the right-occiput transverse orientation at the start of the simulation, as seen on the graph in Figure. 5.29.

The behaviour seen on Figure 5.29, where the head is arrested seems anomalous. The head seems to arbitrarily make contact with the right ischial spine and fails to negotiate around it. The hard ligaments seem to prevent the head from negotiating the spine. Introducing softer ligaments of 100kPa Young's modulus

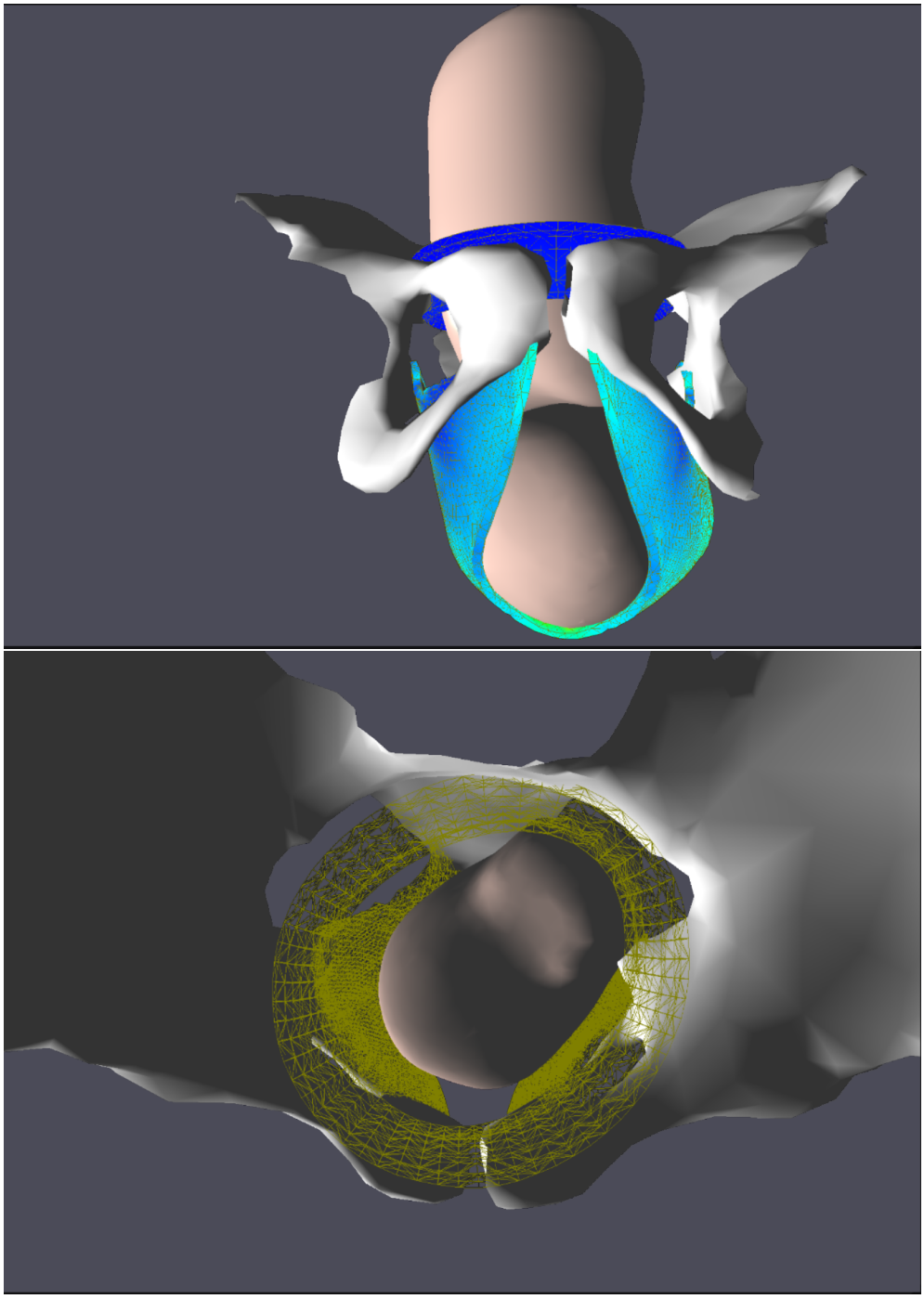


Figure 5.24: Fetal head immediately after it negotiates the birth canal with the bony pelvis and the pelvic floor present. The frontal view of the pelvis (top) shows the pelvic floor undergoing extreme stretching while the head fails to engage the opening. Top view of the pelvis (bottom) shows the head failing to fully internally rotate.

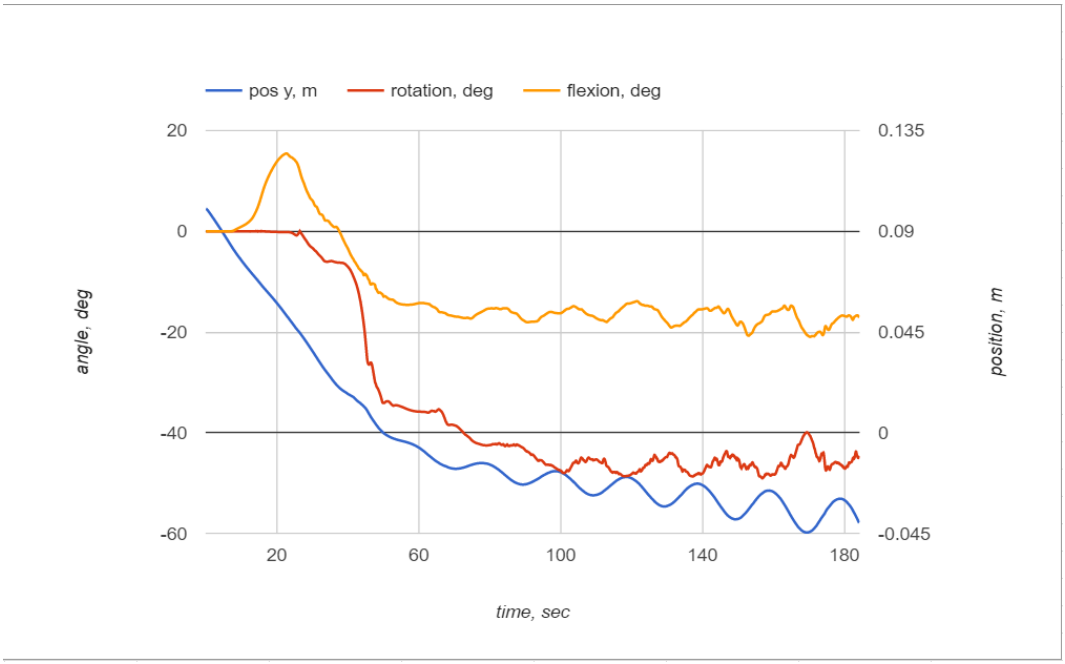


Figure 5.25: Graph of fetal head motion with bony pelvis and pelvic floor present. The oscillation in the end show the head being suspended on the pelvic floor and unable to descend further while being under the effect of the pulsating uterine explusion force.

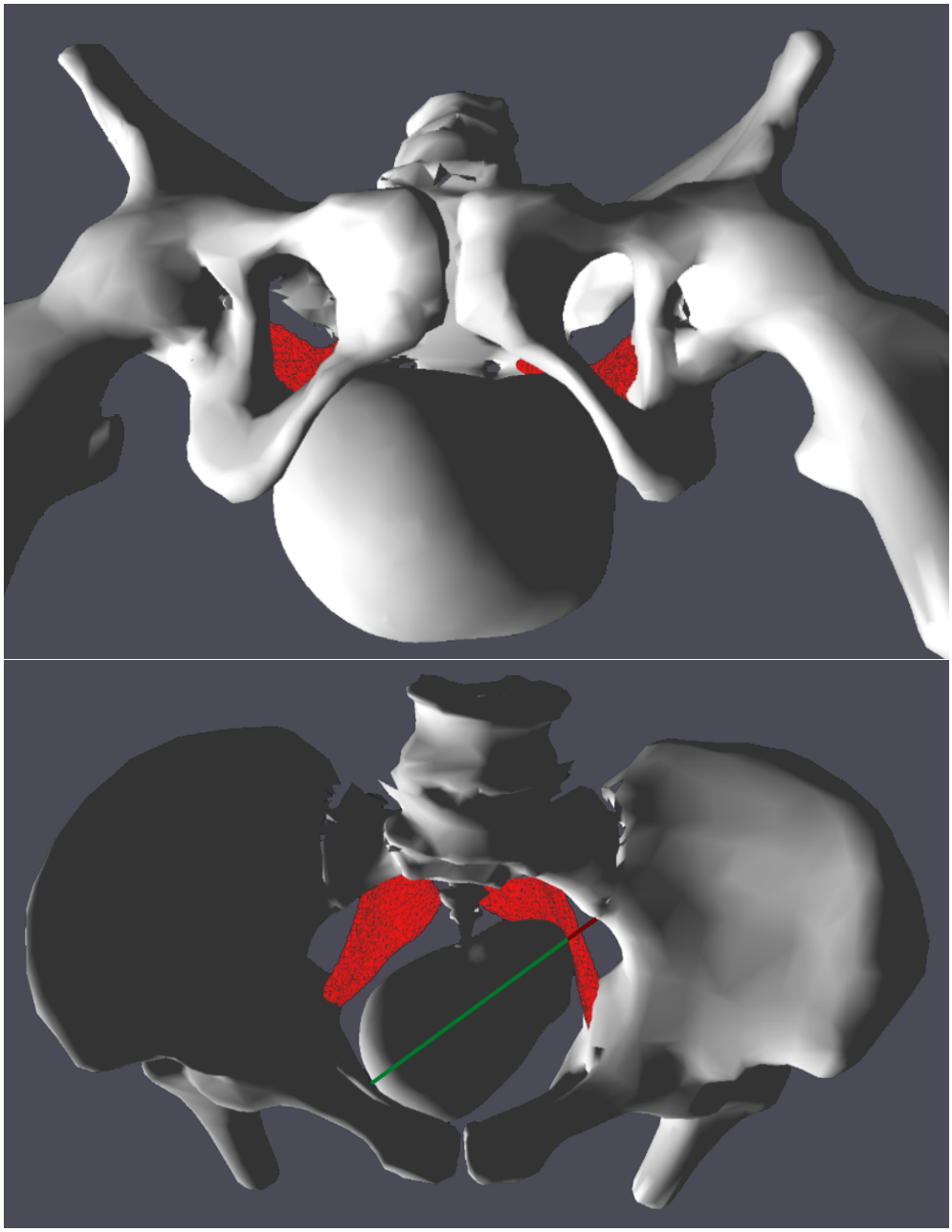


Figure 5.26: The orientation of the skull after negotiating the birth canal when only the rigid bony pelvis and the sacrospinous ligaments are present. Note that the skull is pushed away from the ligaments (bottom image) so that the occiput touches the pelvis on the opposite side.

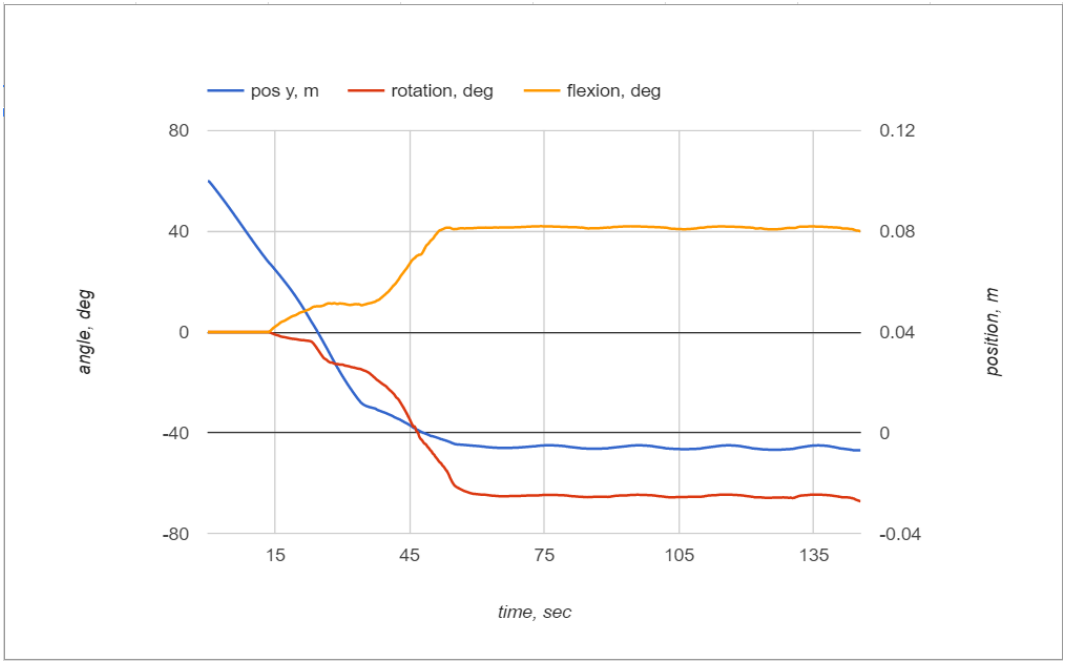


Figure 5.27: Graph of fetal head motion with bony pelvis and sacrospinous ligaments present. The head's descent is arrested with all rotations stopped.

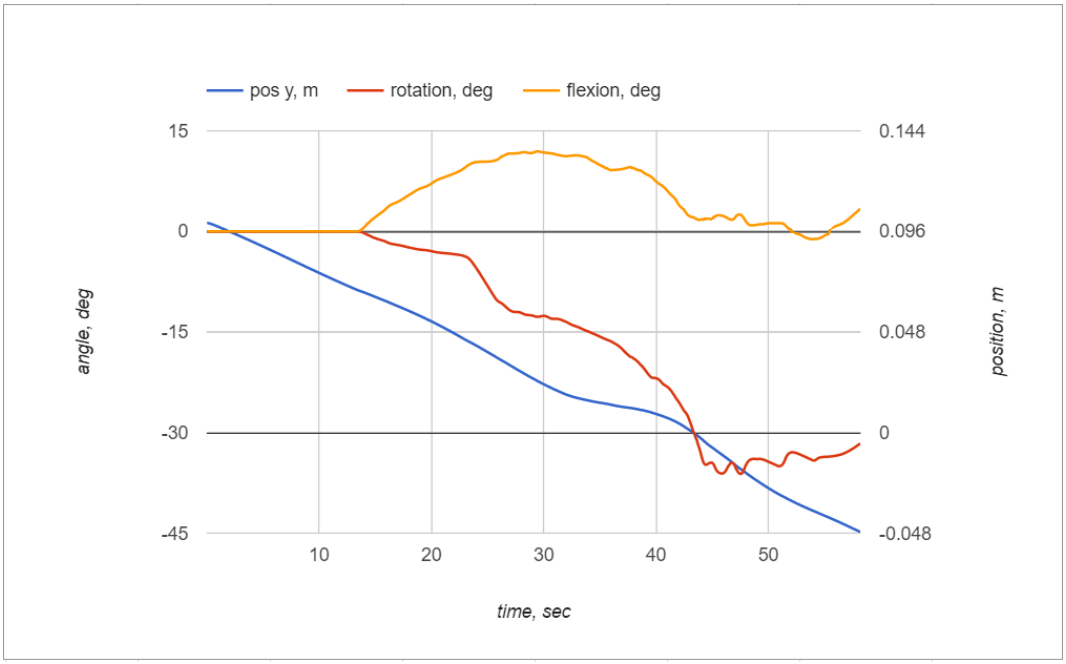


Figure 5.28: Graph of fetal head motion with bony pelvis and sacrospinous ligaments present. Softer ligaments (100kPa stiffness) do not impede the head's descent.

solves the issue as seen on Figure 5.28.

Note that it is not the ligament that interacts directly with the fetal head, but rather it stops the coccygeus muscle from "falling into" the greater sciatic foramen.

Bony pelvis with all soft tissues

As described above the introduction of ligaments alone did not yield any additional cardinal movements, nor did the addition of the pelvic floor on its own. However, when all three components are present in the simulation, further movements were observed.

As the rigidity of the ligaments increases so does the rate at which the internal rotation occurs. Softer ligament stiffness values (less than 60kPa Young's modulus) fail to provide enough support for the head and prevent it from entering the greater sciatic foramen. The head does not fully engage with the foramen but

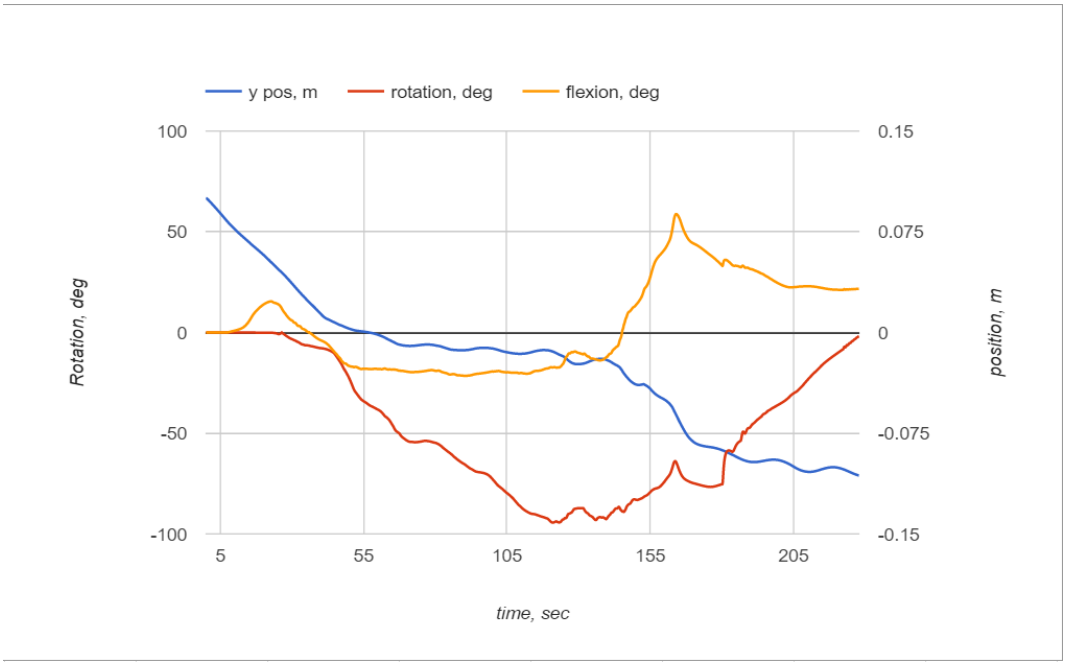


Figure 5.29: Graph of fetal head motion with bony pelvis, cervix, pelvic floor and sacrospinous ligaments present. The head can be seen undertaking all cardinal movements. The flexion (descent of the orange curve) and the extension (the peak of the orange curve at around 155 sec) are shown. The internal rotation is shown gradually reaching the full value of 90 degrees (red curve going down). The external rotation (red curve returns to the original value of 0) occurrs after extension (peak on orange curve) and further descent of the head (blue curve going down).

the rotation is slowed down. With the rigidity of the ligaments reaching higher values (more than 1MPa Young's modulus) the rotation is observed more consistently and the rotation rate increases. We do not evaluate the actual rotation rates as the total childbirth simulation times of approximately 300 seconds do not correspond to the duration of typical second stage of labour. In fact for a real-life childbirth the ligaments are considerably more stiff to stop the head from engaging into the foramen. The experiment was only conducted to check whether or not the ligament stiffness affects the internal rotation rate and it does!

Figure 5.29 demonstrates the trajectory of the head when the pelvic floor, cervix and the ligaments are present.

Table 5.7: Effects of adding maternal pelvic organs on the occurrence of the internal rotation.

Pelvis	Ligaments	Pelvic floor	Internal rotation	Delivery
Yes	No	No	Minor	Yes
Yes	Yes	No	Minor	Yes
Yes	No	Yes	Minor	No
Yes	Yes	Yes	Full	Yes

The middle range of the graph shows how the descent of the fetal head is halted (flattening of the positional curve), while the internal rotation is occurring. Upon the completion of the internal rotation (seen as the rotation curve reaching around -90 degrees) the head progresses further. The rotation can be seen oscillating around the 90 degree mark indicating pulsating intra-uterine pressure.

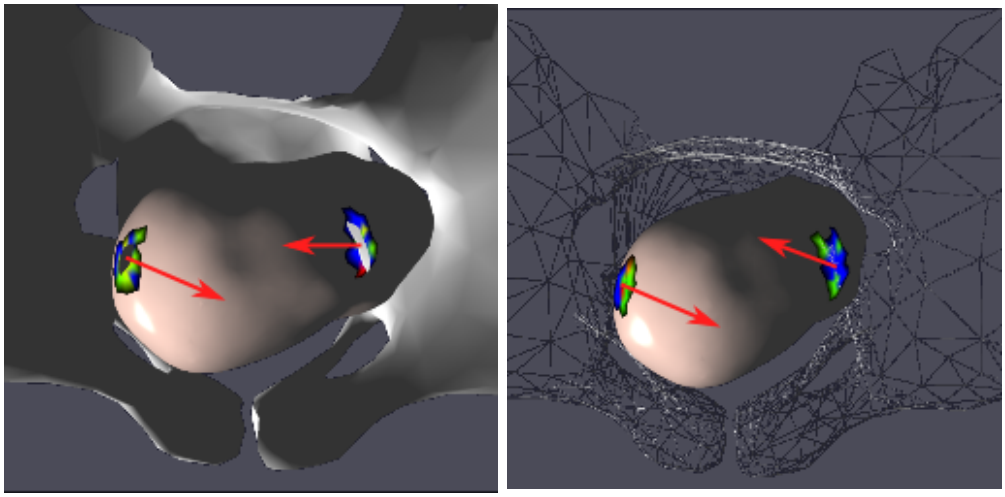
The return of the rotation curve from the lower values towards zero indicates the occurrence of the external rotation, whereby the head exhibits restitution to its initial orientation.

The flexion curve shows slight extension in the very beginning, due to misalignment of the initial fetal head position and the pelvic inlet. It is, however, mostly irrelevant. Further change in the curve indicates the beginning of the flexion. Note how the flexion and the internal rotation overlap initially. The flexion reaches the value of approximately 20 degrees and remains there for the majority of the simulation. The sharp spike in the flexion curve at the end of the graph is the extension that occurs rapidly as the head emerges through the birth canal (around 155 seconds).

Summary of structures required for internal rotation

The various outcomes described above are summarized in Table 5.7.

The main finding here is that the pelvic floor is required for pivoting while the head rotates. The ischial spines are the main initiators and the main contributors to the torque rotating the head. The ligaments are also important as they prevent the head from engaging into the greater sciatic foramen, where it "gets stuck". Removing any of these three contributing factors leads to the failure of consistently and fully observing the internal rotation.



(a) (b)

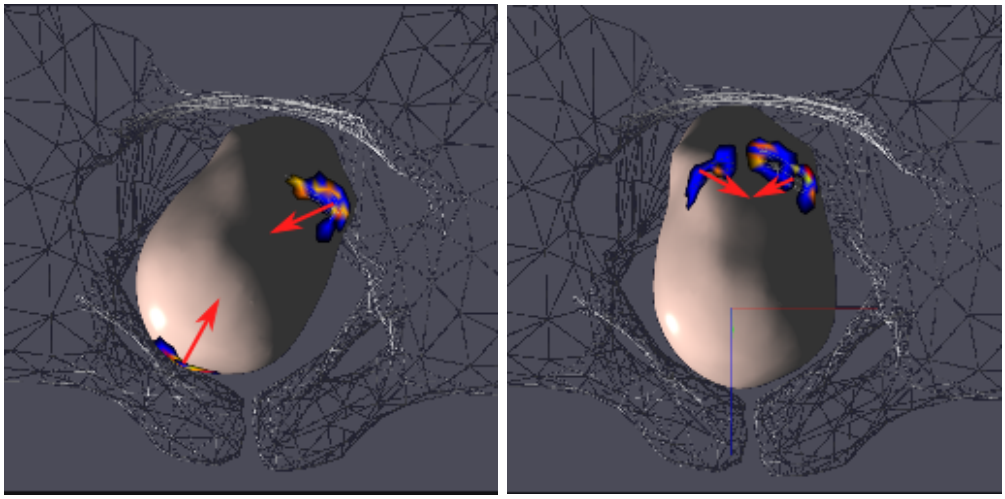
Figure 5.30: The first half of the internal rotation. Ischial spines have a major contribution towards the rotational torque.

Mechanics

The contact of the fetal head with the the ischial spines, sacrospinous ligaments and the pelvic floor has been shown to have a major effect on the internal rotation. The actual origins of the torque generating the rotation can be seen visualized in the next three figures depicting different stages of the internal rotation. Figure. 5.30 shows the initialization of the rotation due to the asymmetric contact between the head and the ischial spines.

During the second half of the internal rotation the ischial spines are no longer contributing to the rotational torque. Most of the torque is coming from the pelvic floor muscles and the ligaments. This is illustrated in Figure 5.31.

Figure 5.31b shows the final stage of the internal rotation where the head is rotated anteroposteriorly. The rotation stops mostly due to the fact that the asymmetry in the contact forces is no longer present. The figure shows that the coccyx contact areas are mostly symmetrical.



(a) (b)

Figure 5.31: The second half of the internal rotation. Sacrospinous ligaments and the pelvic floor have a major effect on the rotation. The rotation is complete when the coccyx is reached and the contact forces are symmetrical.

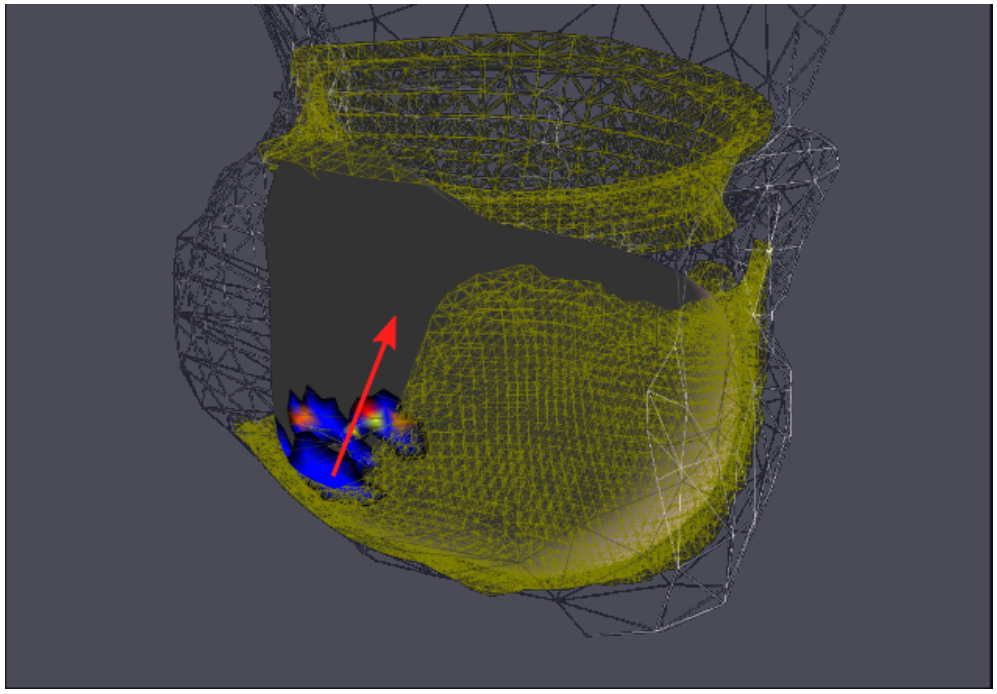
5.5.3 Simulated extension

The extension was successfully simulated subject to sufficient resistance occurring during contact with the coccyx area of the maternal pelvis. The normal of the contact surface between the fetal skull and the maternal coccyx is pointing anteriorly towards the maternal front. This is the direction in which the head must move for the extension to occur.

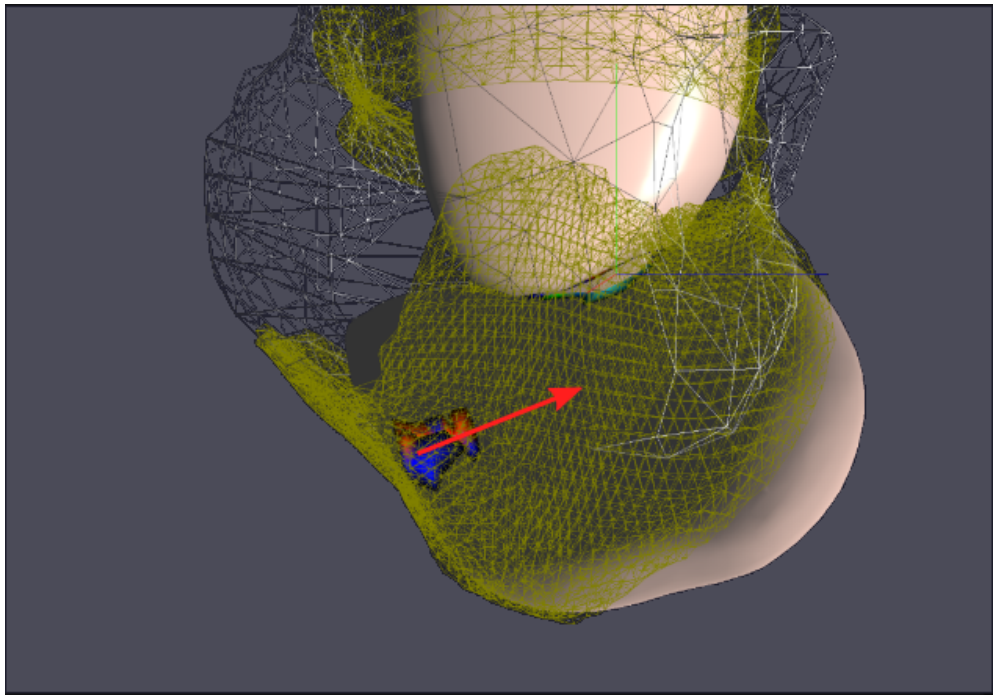
Mechanics

The flexion is maintained until the extension occurs as the head descends. The descent moves the contact point towards the fetal forehead offsetting the contact force to act more anteriorly as opposed to pointing upwards. This is illustrated in Figure 5.32.

Two models of the maternal pelvis were used to perform different experiments. The anatomically typical pelvic 3D model will have a distinct curve from the inferiorly pointing pelvic inlet axis. The original model with the prominent coccyx present was used and provided sufficient push on the fetal head to initiate the extension. The second mesh was modified to decrease the forward protrusion of



(a)



(b)

Figure 5.32: Extension initiated due to the change in the direction of the contact force between the head and coccygeal area as the head descends.

the coccyx. The direction of coccyx was changed to follow the anterior plane of the maternal sacrum, with the coccyx pointing downwards instead of being slightly directed anteriorly. Without the initial push forwards the contact forces between the head and pelvic floor become the main contribution to the rotation of the head in the stage when the extension should occur. The extending rotation requires torque and the contact with the pelvic floor alone provides insufficient torque for the extending rotation to occur. The contact forces are mainly pointing upwards directly against the expulsion force thus only slowing down the descent of the head, but fail to apply any torque rotating the head anteriorly. It could be described as the head being suspended in the hammock made of pelvic floor muscles, as seen in Figure 5.33.

Notably, after the initial redirection the coccyx seems to have little effect on extension. The contact areas (intersecting triangles) are mostly absent when compared to the period of extension initiation when the coccyx is giving the head the initial push. Most of the force for increasing extension maintaining extension after the initial push is coming from the perineal area of the pelvic floor which is pushing the fetal forehead thus exerting force superiorly and anteriorly.

The crowning during delivery is the phenomenon when the fetal head is clearly visible through the vaginal opening. It is also notable that when crowning is established the fetal head will not typically slip backwards in the opposite direction of being born. The simulated crowning also follows the same characteristics. It could be argued that the non-retreating state of the head during crowning is due to the frictional hold of the pelvic floor and vaginal surfaces acting on the head from the sides. As previously stated, our simulation models frictionless contact and if the above statement were true then it should not be possible to simulate non-retreating crowning. Based on this it can be argued that friction between the fetal head and the tissues near the vaginal opening is not the only or even the main factor on preventing the slippage of the head backwards. The main factor contributing to head's stable stage appears to be the direction of the contact forces between the head and the said soft tissues. As seen on Fig. 5.34

The peak of the extension is reached during the crowning when the perineal region of the pelvic floor is in contact with the fetal face in the maxillary region where the head is most protruding. After the fetal chin passes through the vaginal

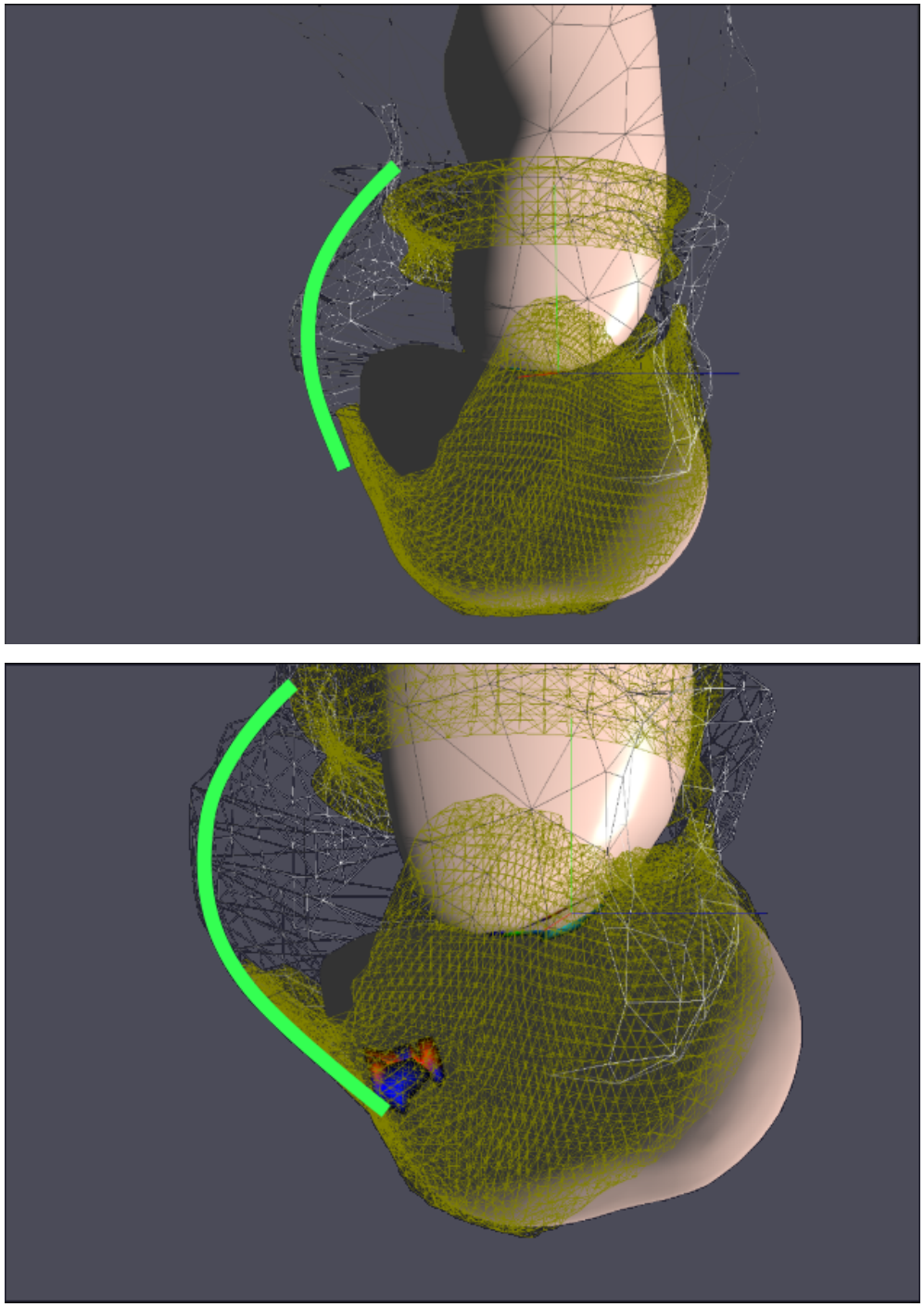


Figure 5.33: Sagittal view of the maternal pelvis and fetal head during normal labour at the stage of extension. The first pelvis is modified to flatten the sacrum by moving the coccyx backwards thus reducing the curvature of the pelvic axis (top). The fetal head descends without extending and halts suspended. The bottom image shows a normal pelvis and head undertaking extension from contact with the curved sacrum. The curve of the sacrum is highlighted on both images.

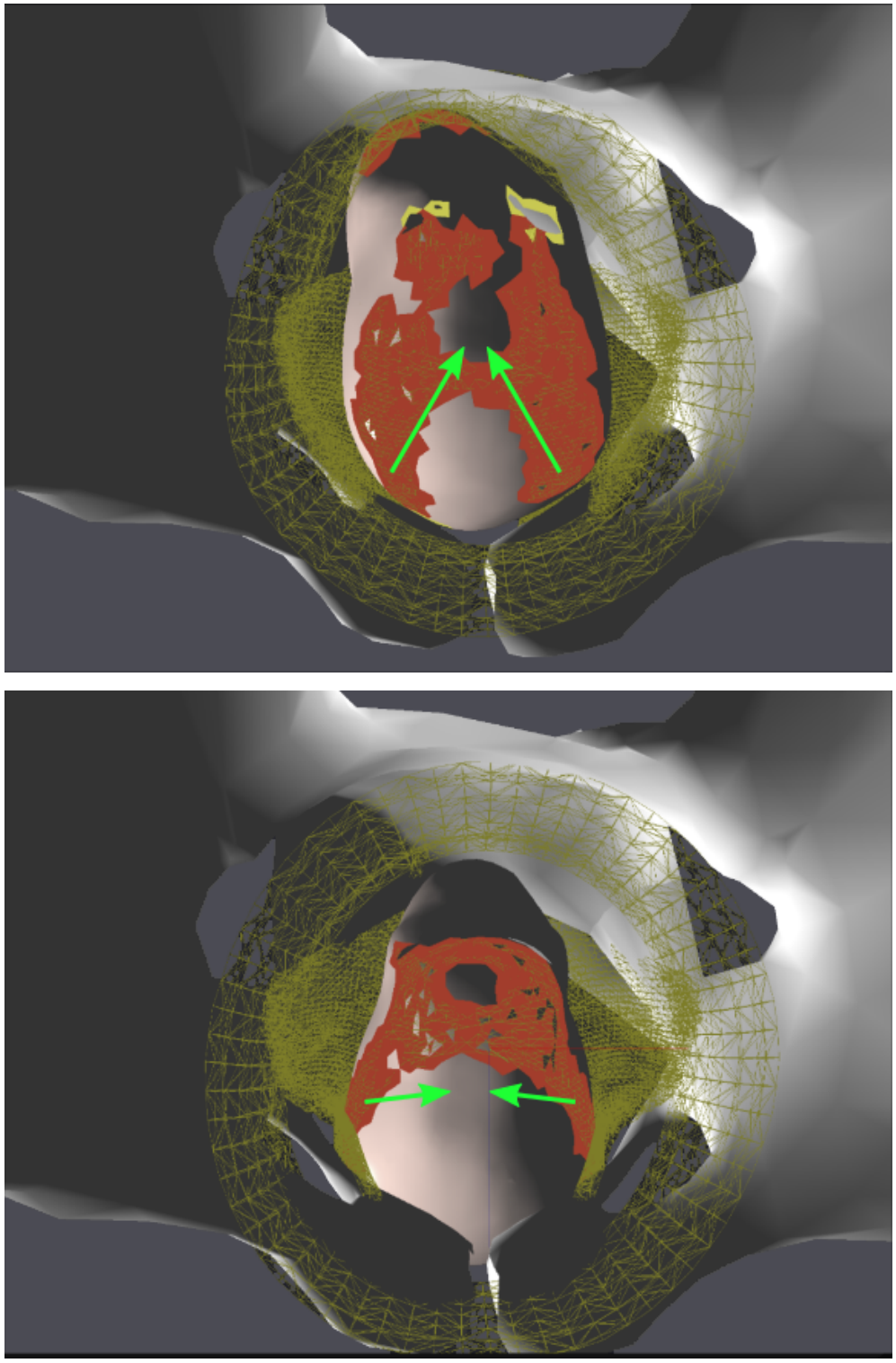


Figure 5.34: Figure showing contact areas (red) and force directions (green arrows) between before (left) and during (right) crowning. Forces direction before crowning are directed posteriorly, whereas during crowning they point transversely in the opposite directions cancelling each other and preventing the head from being pushed back.

opening the contact force acting on the bottom of the head decreases which leads to a noticeable decrease in the magnitude of the extension.

The presence of such phenomena closely correlates to real labour thus further suggesting that our approximation is valid, although not perfectly accurate. The inaccuracy comes from the lack of a proper neck model in the simulation setup as the fetal neck is represented by an infinitely thin spring of high stiffness. The spring has no collision shape and will not interact with any other object in the simulation apart from the objects to which its ends are attached. This leads to the decrease of extension being excessive as there is no neck to stop the head from un-extending further.

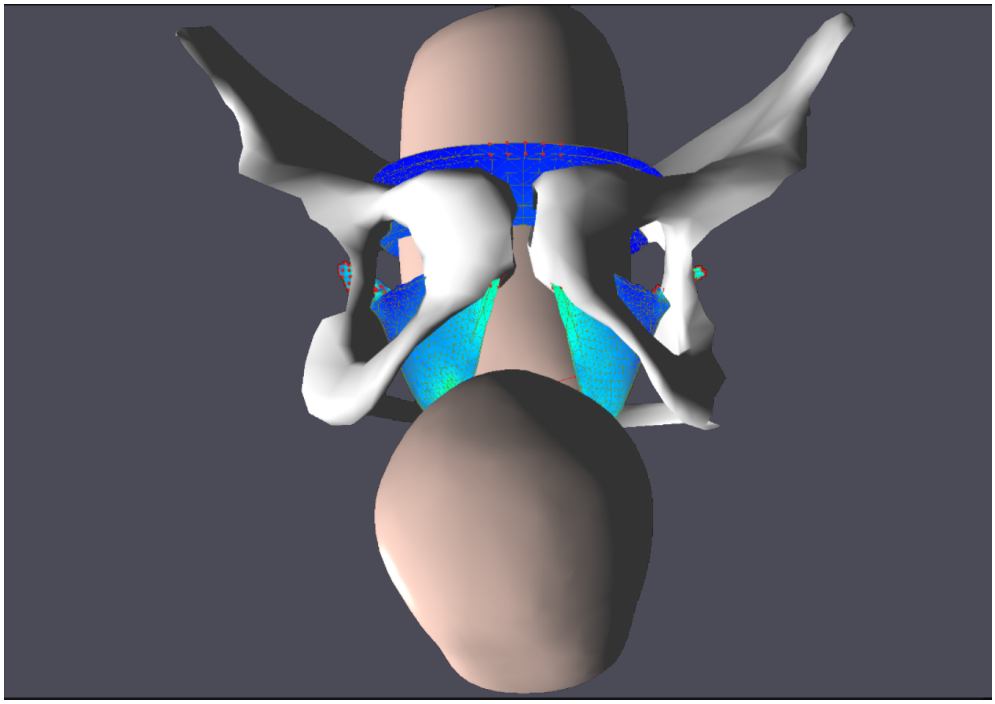
5.5.4 Rotation of fetal shouders and simulated external rotation

The experiments investigating the occurrence of the external rotation and the fetal shoulder rotation are correlated. The former rotation is directly caused by the second rotation of the fetal shoulders. Both of these rotations are observed during the simulated childbirth.

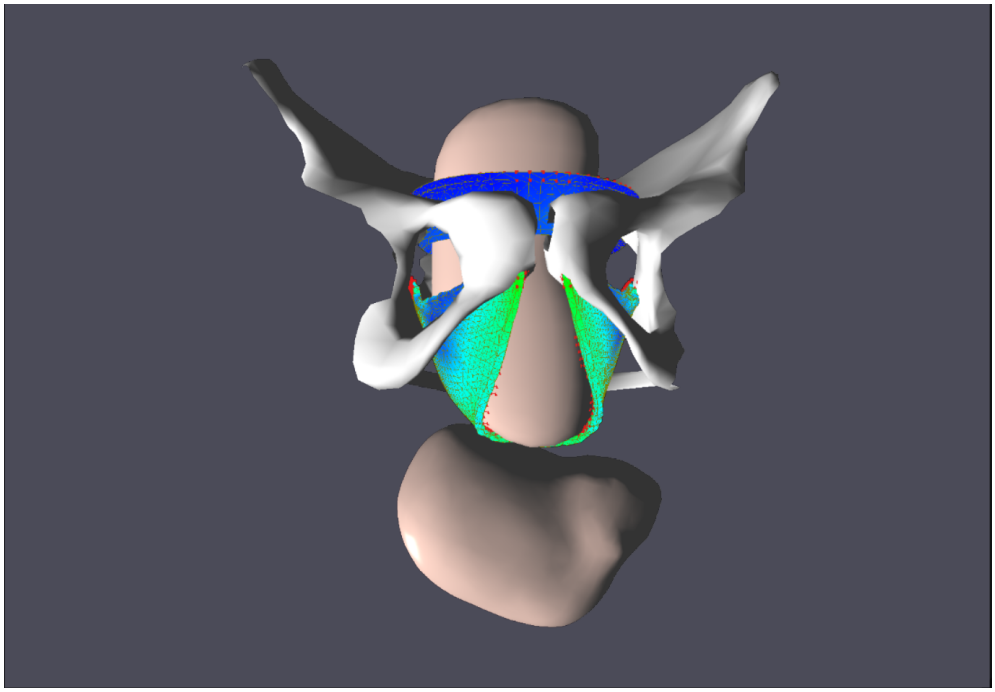
External rotation is defined as the rotation of the fetal head after it exits the birth canal. Simulation results indicate that the external rotation occurs due to the contact between the pelvic floor and the fetal shoulders. It was also observed that the simple trunk and neck models used in the simulation appeared to be sufficient to model the external rotation to a realistic degree

The result of a successful simulation of external rotation is presented in Figure. 5.35. The head can be seen transversely oriented after having been born in the anteroposterior orientation.

The cardinal movement is observed when the fetal shoulders engage with the pelvic floor and slowly rotate. The rotation can be observed occurring towards the maternal left and right. The direction in which the head externally rotates seems to be based on the initial minor offset of the shoulders from the transverse orientation. If the shoulders engage with the pelvic floor with a slight orientation offset relative to the vertical axis towards the maternal left, the angle will increase in the same direction.



(a) Head is born in the anteroposterior orientation.



(b) Trunk internally rotates and causes the head to rotate externally.

Figure 5.35: Fetal head undertaking external rotation.

Mechanics

No contact with the bony pelvis or the pelvic floor is occurring by the time the head is undertaking external rotation. Therefore, the external rotation is not caused by the interactions of the head with any of the maternal structures. Rather, the rotation occurs as a result of the torsional stress in the fetal neck. The torsion is caused by the orientation between the trunk and the head. As the trunk undergoes the internal rotation, the head still remains in the anteroposterior orientation as seen in Figure 5.35b. The information on the torsion spring used to simulate the neck and its effect on external rotation is provided in Section 5.3.1.

It should be noted that the direction of the torso is inconsistent with real cases of childbirth. The orientation of the head is vertically downwards. The inflexibility of the simulated torso prevents it from undergoing extension whereby the shoulder area of the torso is bent anteriorly towards the maternal front. Our rigid fetal torso is unable to deform in the way that would facilitate such configuration. This limitation is expressed in the Future works section 6.3.

5.5.5 Adverse presentations

Brow presentation

The required delivery force for the successful simulated delivery was greatly increased when the head was initially extended mimicking brow presentation. The original transverse orientation of the head was preserved, but an extension of 45 degrees was pre-imposed. This is not necessarily accurate, as the brow presentation can occur during labour and not be pre-imposed. It was, however, deemed of interest to study the effects of such presentation on the occurrence (or absence) of the cardinal movements. The initial orientation of the head can be seen on Figure 5.36.

Figure 5.37 illustrates the trajectory and rotation of the head during the simulation. It can be seen that the duration of labour is substantially increased. The total duration of labour took almost three times the time taken by simulations of normal presentation. The head can be seen stagnant for a long duration while the expulsion force grows to the required delivery force of 200N at which the birth

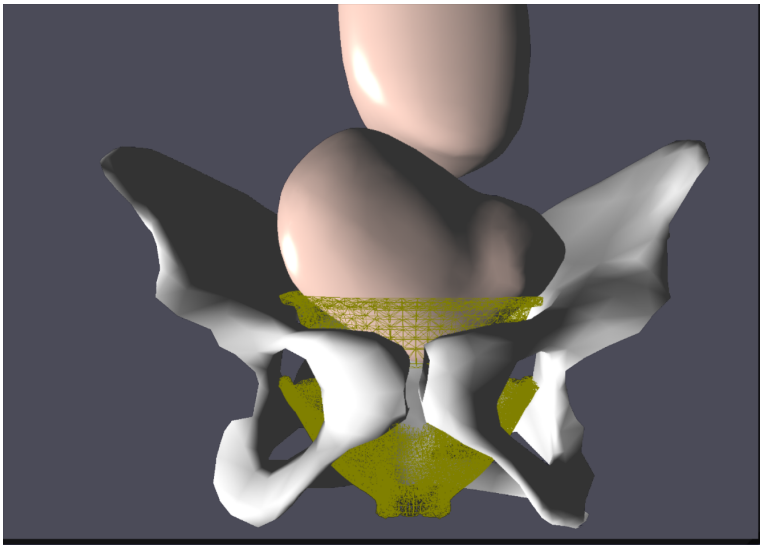


Figure 5.36: Simulated brow presentation setup. The head is extended by 45 degrees to simulate brow presentation.

finally succeeds.

The internal rotation can be seen interrupted at around the 60 degrees mark.

Note that the birth only occurred after the head un-extended and became flexed taking a similar orientation as after the internal rotation in a normal simulation. After the flexion occurred it can be seen that the head completes the internal rotation reaching the required angle of around 90 degrees. This finding further suggests that the internal rotation is dependent on a successful flexion.

Effects of over-flexing

The original mesh obtained from Lapeer and Prager (2001) did not have a mandible. When the simulation was run with this mesh, the head was able to flex to 60 degrees. The view of the head as it is undergoing such flexion is shown in Figure 5.38.

When the mandible was added to the skull, the maximum flexion was reduced. The mandible engaged the fetal chest area and the flexion reached 45 degrees.

The simulation resulted in flexion, internal rotation, but then failed to extend. The extension was found to depend on the contact of the coccyx and the fetal forehead. The major flexion prevented this from happening, as the coccyx is

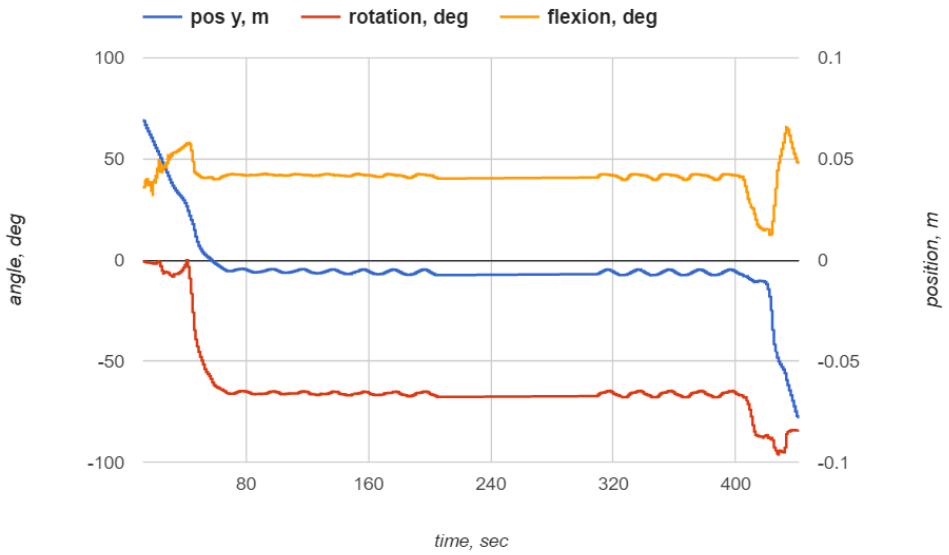


Figure 5.37: Graph of the rotation, flexion and the vertical position of the fetal head during a simulation of brow presentation.

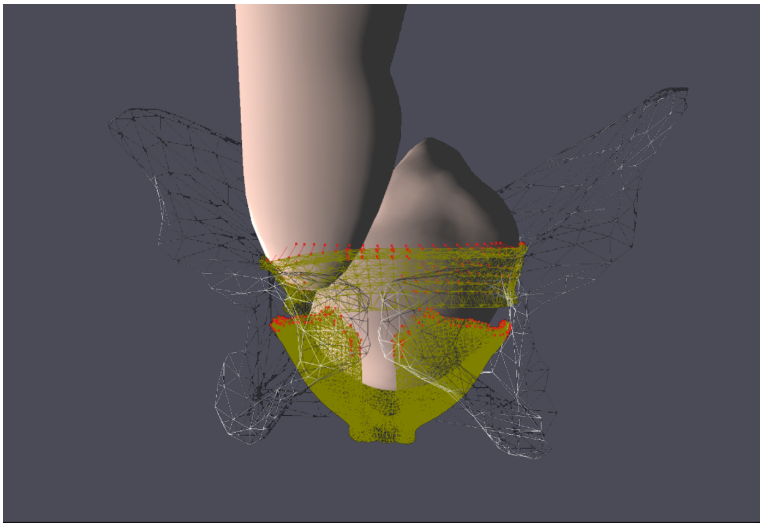


Figure 5.38: The fetal head undergoing a flexion of 60 degrees.

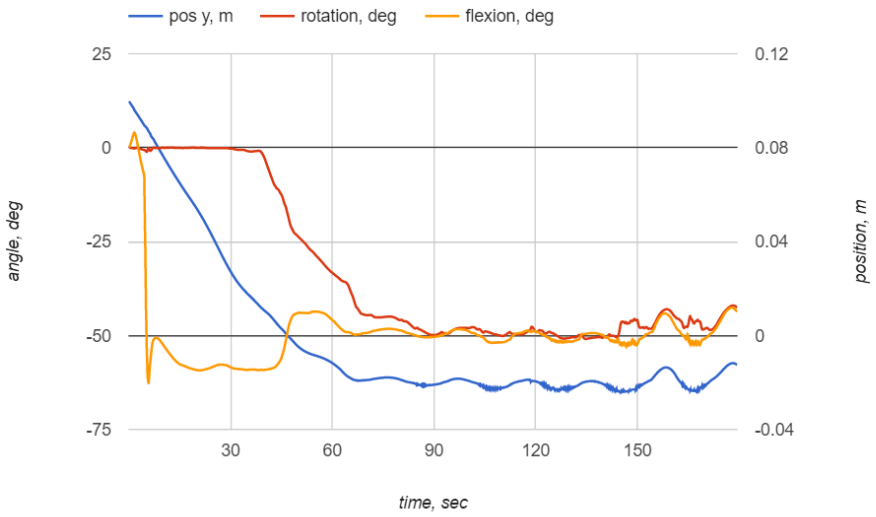


Figure 5.39: The simulation of the fetal head without a mandible with major flexion pre-imposed. The simulation fails to complete as the blue vertical position curve can be seen oscillating around a fixed value.

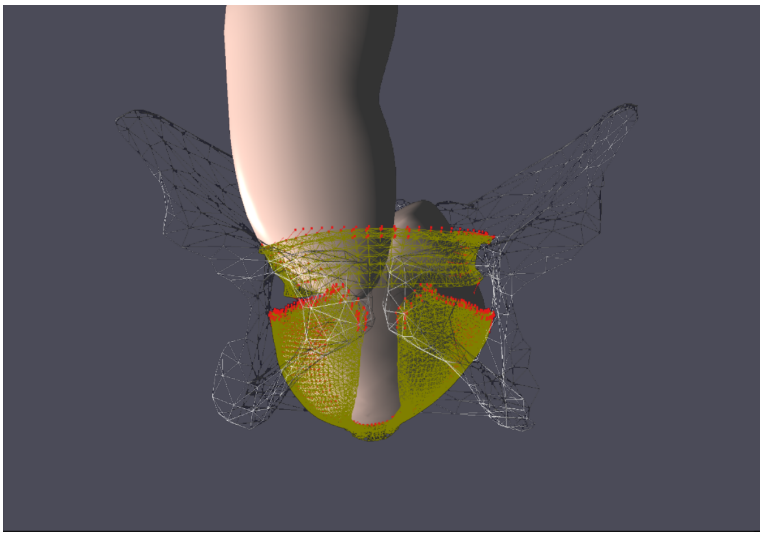


Figure 5.40: The fetal head undergoing flexion of 45 degrees.

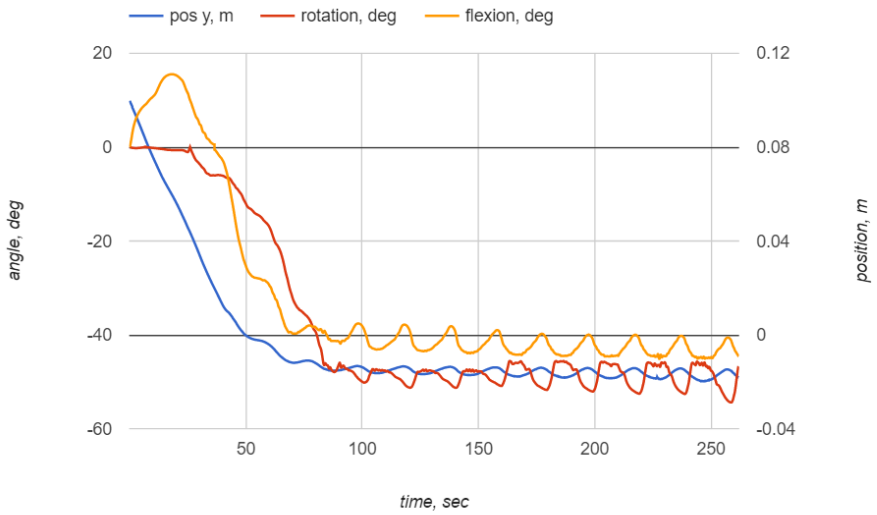


Figure 5.41: The head's trajectory when a 45 degree flexion was allowed.

pushing on the tip of the head near the anterior fontanelle. The contribution of the pelvic floor to the extension was also prevented. In fact the pelvic floor was contributing to the flexion rather than the extension as it pushed against the occiput of the flexed head.

The over-flexing could have an effect on the internal rotation. The fully flexed head that engages with the middle segment of the pelvic cavity will have little interaction with the ischial spines and the pelvic floor. This results in an inhibited internal rotation. The lack of proper internal rotation leads to further complications such as prolonged simulated labour duration and complete arrest of the head.

Only after the mandible was added and the fetal chest area was pulled forward to prevent the over-flexion the simulation resulted in success. The results of these successful simulations were presented in the previous sections.

5.5.6 Material properties variation

The material properties of the maternal pelvic floor were changed as part of an experiment to establish if the particular values of pelvic floor muscle stiffness

Table 5.8: The material property values used in the experiment and force F (N) required for the delivery which had a duration of T (sec).

multiplier	u	K	F	T
0.5	33000	500000	70	180
1	66000	1000000	150	300
2.5	165000	2500000	250	650

would have a major effect on the simulation. The main aim of the experiment was to determine if the cardinal movements will occur with softer or stiffer pelvic floor muscles.

The material properties used in the previous simulation were 66 kPa for the shear and 1MPa for the bulk modulus. Raising the values of the moduli up to 165kPa and 25MPa did not cause the cardinal movements to stop occurring. The cardinal movements were observed in a consistent manner. However, a major effect was observed on the duration of labour and the maximum required force for the delivery. The lower values of 33kPa and 500kPa for the shear and the bulk moduli did not prevent cardinal movements either, but showed major reduction in the time duration of the simulated labour and reduced the required delivery force. The summary of the material properties and their effects on the duration of labour and the required expulsion force is provided in Table 5.8. Fig. 5.42 demonstrates a plot of the observed changes.

5.5.7 Varying fetal head size

The size of the fetal head could be of great importance and have a major effect on the experiments. To study these effects, the fetal head size was varied and simulations were run. The head was resized using uniform scaling, therefore increasing or decreasing all the obstetric diameters. The scaling is defined relative to the original mesh with biparietal diameter of 9.2 cm and supraoccipitomental of 13 cm (see Figure 1.7 for diameter descriptions). The values of scaling are listed in Table 5.9 along with the effects of scaling on the duration of labour and the required delivery force. The trunk was also scaled by the same amount to allow for head-trunk interaction to maintain a realistic amount of flexion.

The forces required for the delivery of the scaled up fetus were much larger

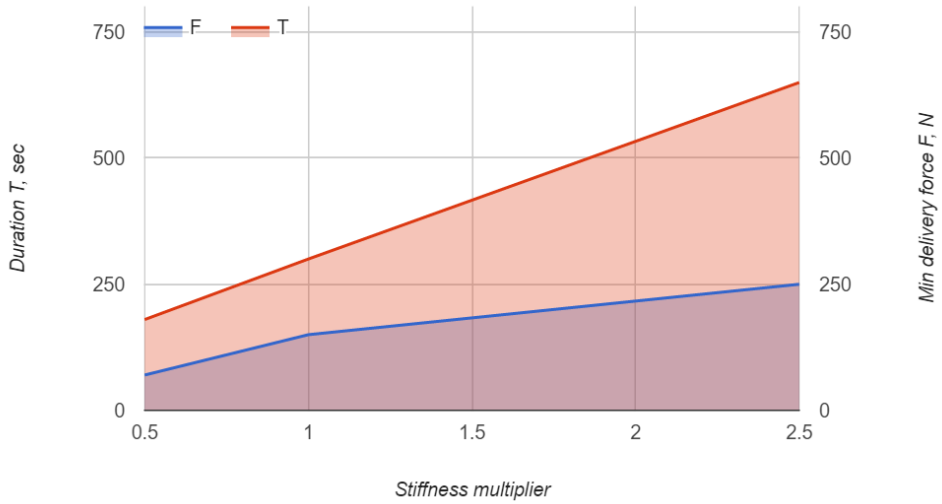


Figure 5.42: A plot showing the effect of varying the material properties of the pelvic floor on the duration of labour. The horizontal magnitude is the multiplier of the stiffness values. T - total time of delivery, F - maximum required force for the delivery to occur.

than the reported maximum values exerted by a typical mother (Ashton-Miller and DeLancey, 2009). Our simulation of the largest head required a peak force of 250N to be born, but only after a simulated ventouse was applied. The internal rotation was complete, but the extension would not be successful even with such a large expulsion force. A pulling force of 20N was artificially applied to the occiput of the head. This pull generated extending torque and helped the head to be born. This artificial force mimics an intervention from an obstetrician. The head could potentially be born without the intervention with higher expulsion forces, but the simulation could become unstable if higher force values were used.

The smallest skull did not engage with the ischial spines to initiate the internal rotation. The head proceeded to descend in the transverse orientation to engage with the pelvic floor. The transverse orientation of the head when engaging with the pelvic floor would not allow the head to be born and it became suspended on the pelvic floor – See Figure 5.43.

The results of combining the experimental outcomes of scaling the fetal head and trunk are presented in Table 5.9. The *cost* measure is added to demonstrate the difficulty of delivering a larger head more consistently. The measure is cal-

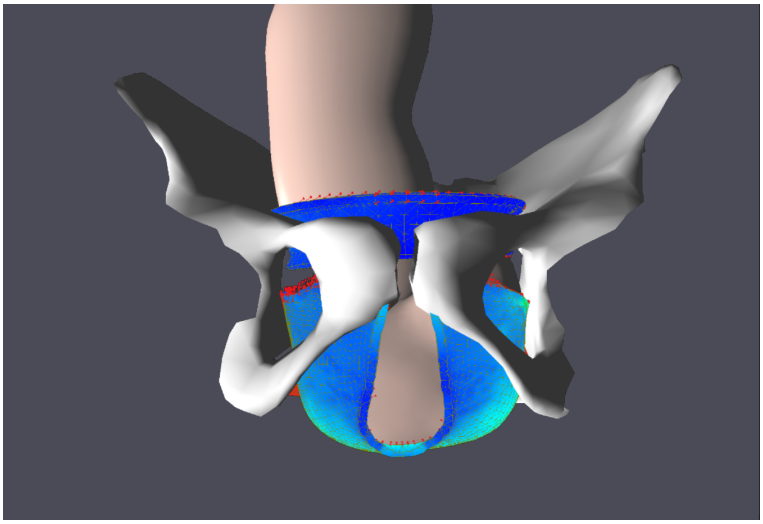


Figure 5.43: Small head failed to internally rotate and became suspended on the pelvic floor.

culated as the product of the delivery force required for birth and the duration of the simulated labour. It can be seen that the duration of labour is not consistently increasing in the graph presented in Figure 5.44, as the increase in the force decreases the labour duration. Their product, however, grows consistently. The exception is the simulation with the smallest head where the *cost* is greater than the larger heads. In fact the normally sized head had the same *cost* as the smallest head simulated. The cause for this behaviour is the fact that the head is presented in the occiput posterior orientation. The occiput posterior orientation is known to prolonge labour (Aishah, 2011).

The graphs for each fetal size as presented in Table 5.9 are shown in the following figures.

5.5.8 Occiput-posterior presentation

The occiput-posterior (OP) presentation was observed during the above experiments with fetal head size variation. The OP features the head presenting with the occiput turned to the maternal back and face turned forward. It was a serendipitous discovery, which occurred when the fetal head size was reduced considerably down to 85% of the original size. The head's trajectory graph is

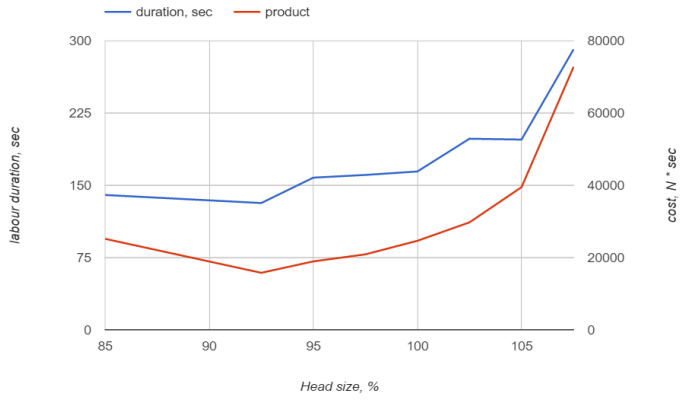


Figure 5.44: Results of scaling the fetalhead and trunk. The *cost* measure combines the duration and the required force for delivery for a more consistent measure. Note how *cost* grows more consistently with the increase of the fetal size.

Table 5.9: Results of varying the fetal scale. The *cost* measure combines the duration and the required force for delivery for a more consistent measure.

scale, %	duration, sec	force, N	cost, N * sec	OA/OP
107.5	291.2	250	72800	OA
105	197.5	200	39500	OA
102.5	198.4	150	29760	OA
100	164.4	150	24660	OA
97.5	160.8	130	20904	OA
95	158	120	18960	OA
92.5	131.65	120	15798	OA
85	140	180	25200	OP

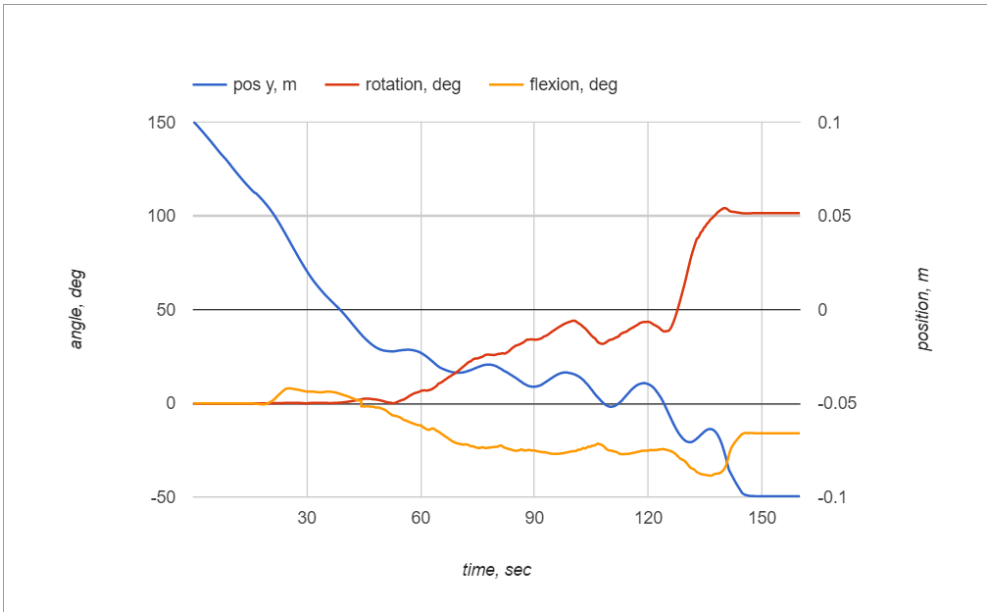


Figure 5.45: Results of simulation with a fetus scaled to 85%.

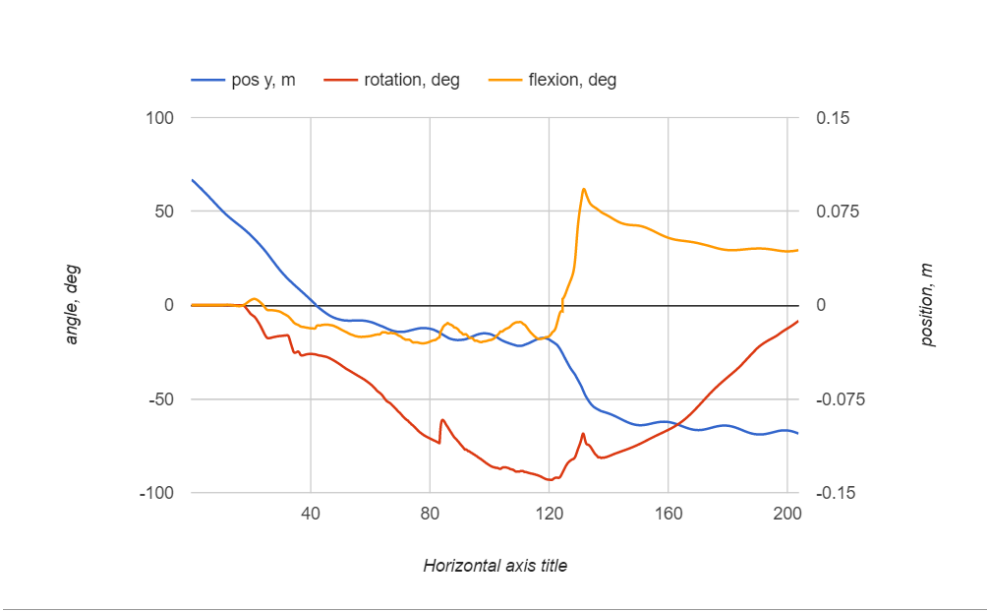


Figure 5.46: Results of simulation a with fetus scaled to 92.5%.

Figure 5.47: Results of simulation with a fetus scaled to 95%.

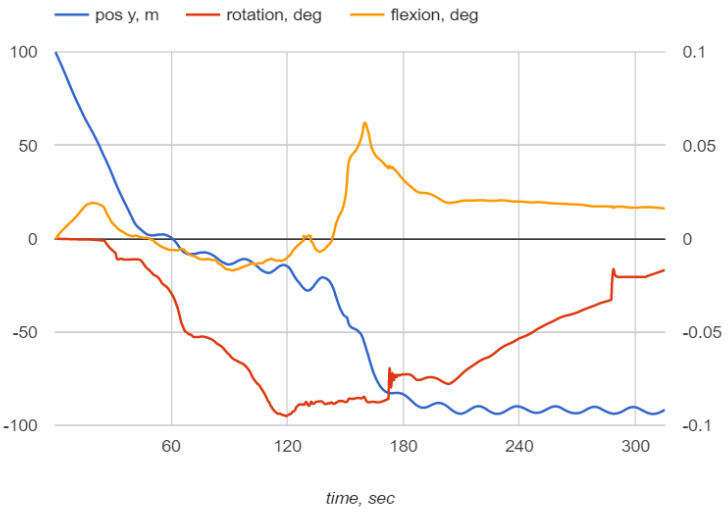


Figure 5.48: Results of simulation with a fetus scaled to 97.5%.

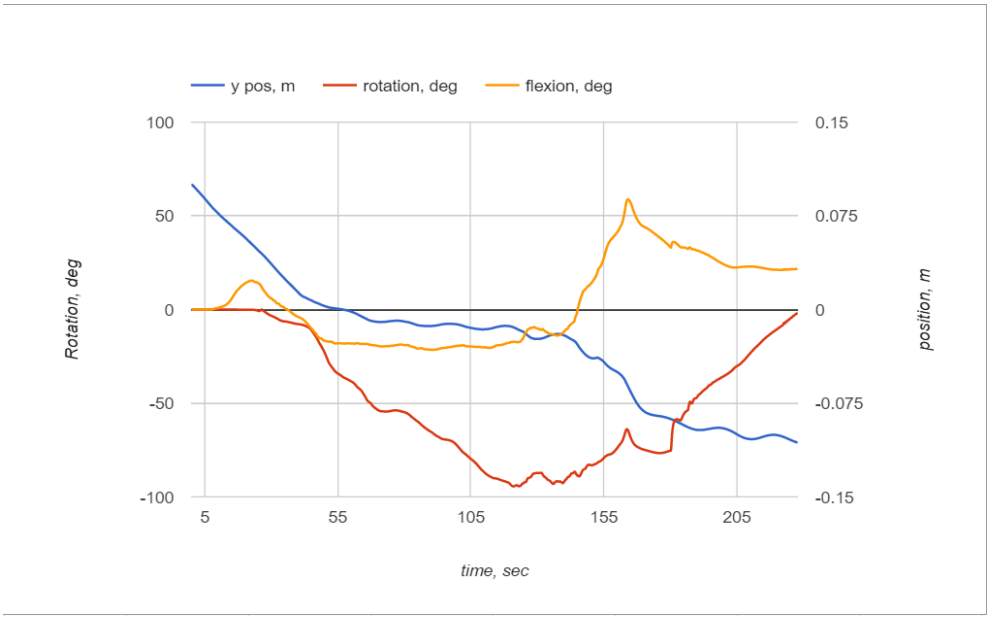


Figure 5.49: Results of simulation with a non-scaled fetus.

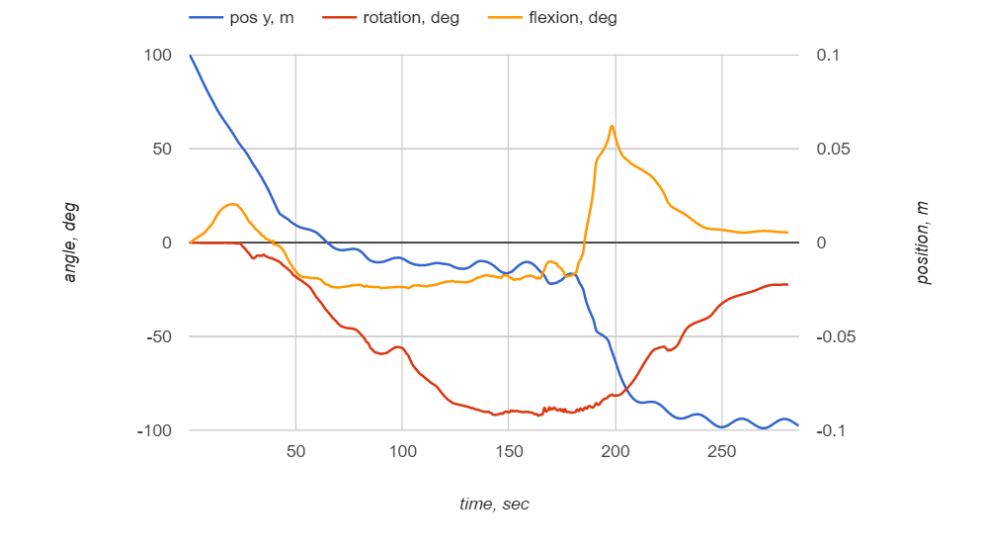


Figure 5.50: Results of simulation with a fetus scaled to 102.5%.

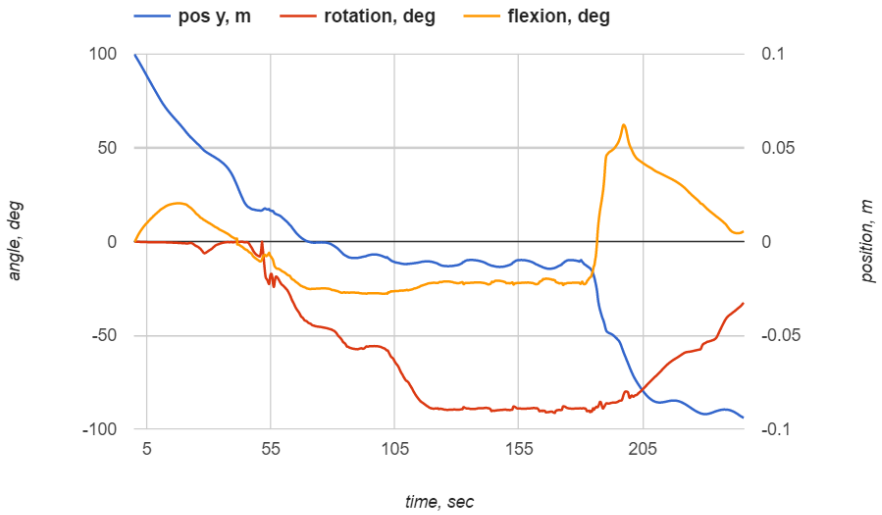


Figure 5.51: Results of simulation with a fetus scaled to 105%.

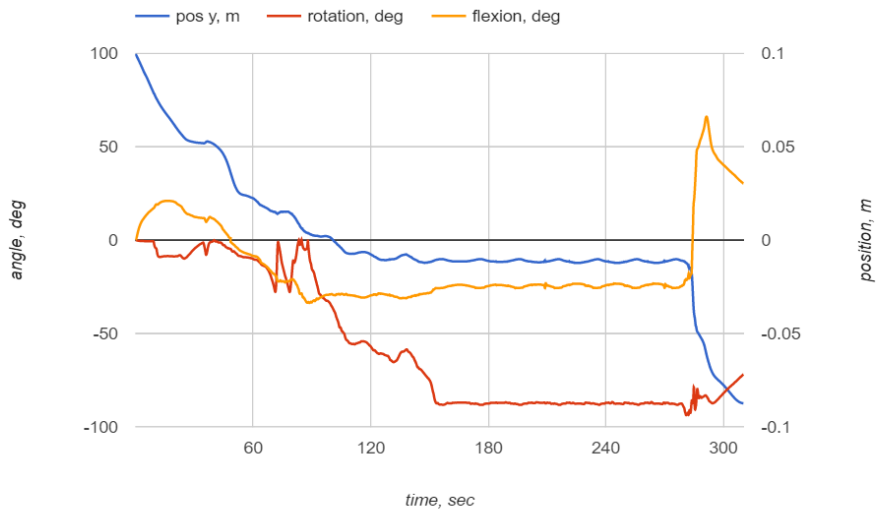
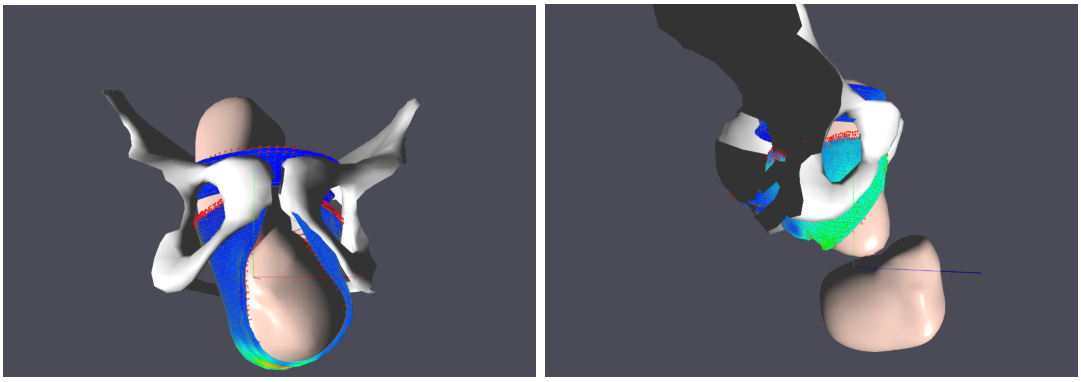


Figure 5.52: Results of simulation with a fetus scaled to 107.5%.



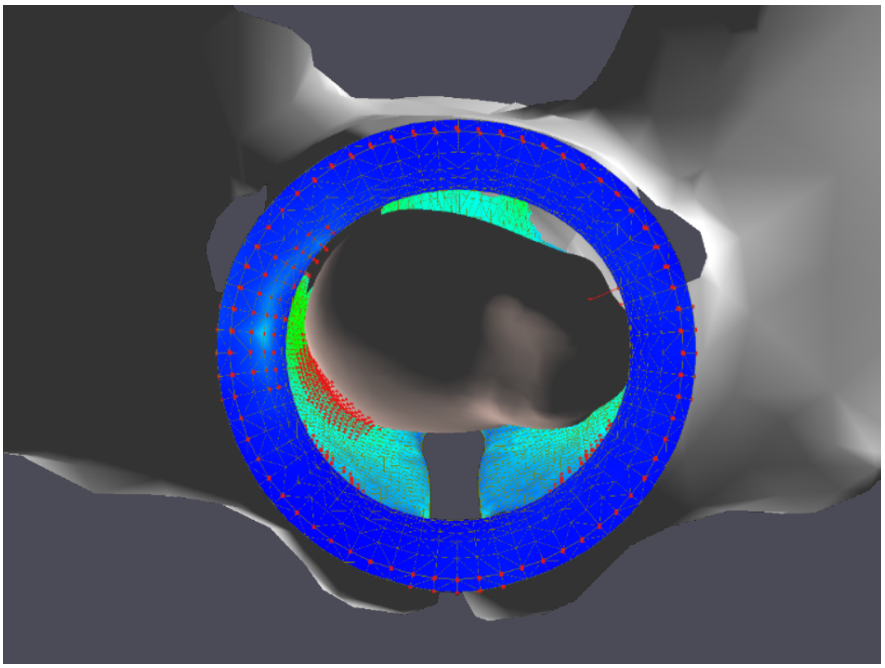
(a) (b)

Figure 5.53: A frontal view (a) of the head scaled down to 85% being born in the occiput-posterior orientation. A side view (b) 10 seconds later where the fetal shoulders are also born.

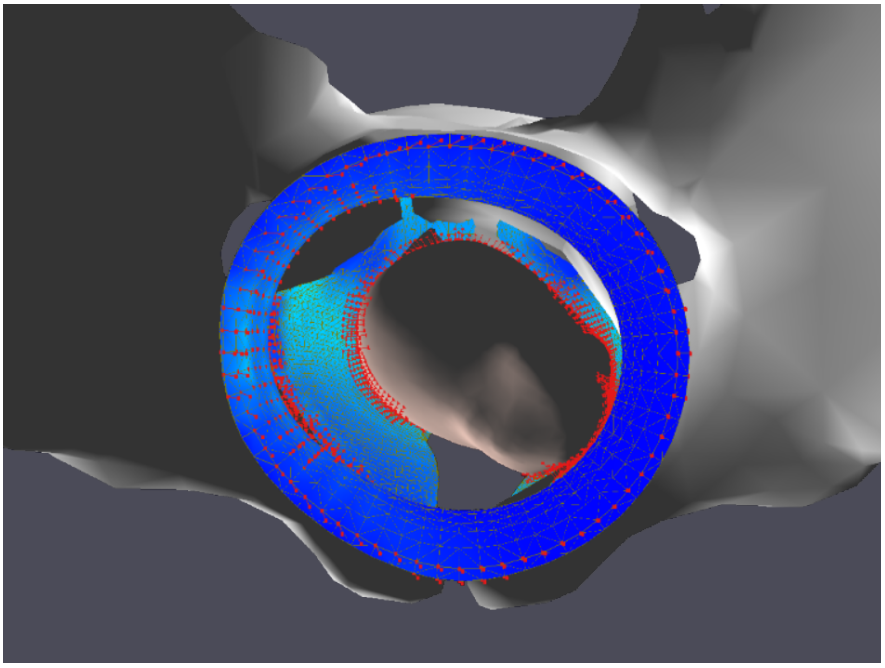
shown in Figure 5.45. The frontal and side views of the head during on OP presentation

The head could be seen rotating anteriorly in the opposite direction of the normal internal rotation, which leads to OP presentation. The superior view of the head as it rotates is shown in Figure 5.54, 5.55. The rotation starts later when compared to the simulations with the original head size. The normal posterior internal rotation is initiated by the ischial spines which are located above the pelvic floor. In the case of the smaller head, the ischial spines do not contribute to the rotation, but rather the interactions with the pelvic floor alone is causing the rotation. Therefore, the head needs to be descended further to engage with the pelvic floor before the internal anterior rotation begins.

Note the large pelvic floor stretching in Figure 5.53. This is in conformance with the clinical observations stating the OP presentation results in larger strain and tissue damage to the maternal pelvic floor (Parente et al., 2010).

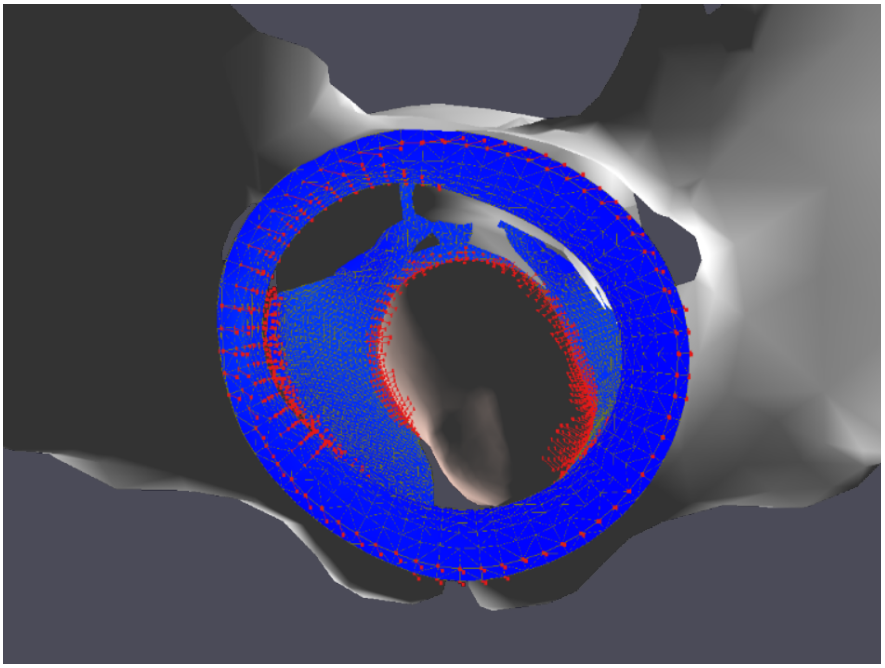


(a)

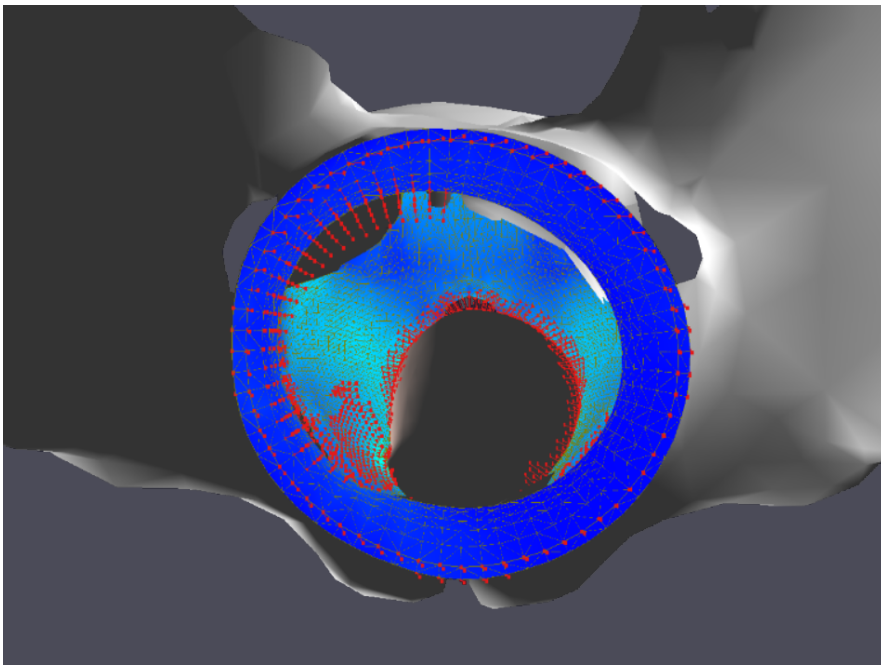


(b)

Figure 5.54: The first half the internal rotation of the head scaled down to 85%. The contact with the pelvic floor causes the head to rotate anteriorly and present in the occiput-posterior orientation.



(a)



(b)

Figure 5.55: The second half of the internal rotation of the head scaled down to 85%. The head further rotates anteriorly.

Chapter 6

Summary and Conclusions

6.1 Conclusions

The objective of this research was to create a childbirth simulation system capable of simulating the cardinal movements during the second stage labour in a forward-engineered manner at interactive simulation rates. In order to achieve this, a FE TLED method was used for the simulation of the soft tissues. A projection based contact method was utilized to model the interaction between the fetus and the pelvic floor. A penalty based contact method was used to simulate the interaction between the rigid fetal head and trunk. The TLED implementation and the contact method were validated to determine their reliability. Then a set of experiments were undertaken to investigate the mechanics of the cardinal movements.

Validation results The experiments conducted to evaluate the accuracy of the TLED implementation and the projection based contact approach were presented in Section 5.2.

A set of experiments were used to compare the results produced by our simulation system and the values produced by Abaqus Explicit. The results showed errors pf approx.0.96% in deformation magnitude between our TLED implementation and Abaqus. The maximum stress errors are considerably higher reaching around 30%, but these extremes are localized. The general distribution of

stress is much more similar between the two implementations. The small errors in deformation suggest that the stress errors have less severe effect. The cube compression experiments further suggest good performance of the implemented contact method. The error of 1.72% is reported between the maximum compression values.

The experiments included a comparison between the projection based contact method and a gap based method based by Heinstein et al. (2000). It was shown that the two methods are equivalent when the simulation time steps are small. With the increase of the time step period the results of the projection based approach were shown to be more consistent. The smaller time steps require significantly more computation so the projection based contact method was deemed to be a more suitable solution for our simulation.

Varying mesh complexity was shown to have little effect on the accuracy of the proposed projection based method.

The performance of the GPU implementation was shown to be superior to the CPU implementation when simulating a mesh of more than 4000 elements.

Observations on the simulation of cardinal movements A series of experiments were undertaken to investigate the mechanisms of the cardinal movements. These experiments were described in Section 5.5. All experiments featured a normal (vertex or OA) presentation.

The *descent* and *engagement* were implemented consistently with all models. These motions required no additional structures except the maternal bony pelvis and the fetal head to be present. The head is seen engaging into the pelvic brim and descending further.

Further motions required more sophisticated models. The findings published previously in (Gerikhanov et al., 2013) were reaffirmed. The results of experiments with only the bony pelvis and the fetus present failed to consistently show internal rotation and failed completely to demonstrate any subsequent cardinal movements.

Introducing the sacrospinous ligaments and the pelvic floor muscles resulted in the occurrence of the internal rotation but only when the fetal trunk was added. The fetal trunk with a basic neck model were necessary to maintain the head's

orientation.

Internal rotation was seen initiated by the pelvic ischial spines, but needed the presence of the pelvic floor muscles for full pivoting to occur. The sacrospinous ligaments were necessary to maintain the internal rotation as it was reaching a complete turn of 90 degrees. Removing any of these three contributing factors lead to the failure to consistently and fully manifest the internal rotation.

The radiological studies by Borell and Fernstrom (1958) have shown the correlation between the fetal head vertical position and its orientation. The majority of cases were shown to have transverse orientation when the head was at the level of the ischial spines. The head assumed a sagittal (anteroposterior) orientation by the time the head was at the lower level of the ischial tuberosities. Our simulation results indicate a similar behaviour. The internal rotation starts at the level of the ischial spines and is complete by the time the head is lower and engaging with the pelvic floor.

The extension was observed and was caused by the contact of the fetal forehead area with the pelvic coccyx. This contact initiated extension and further extension was generated by the pelvic floor.

The external rotation was observed after the delivery of the fetal head. The fetal shoulders engaged the pelvic floor and internally rotated. This in turn caused the head to follow in rotation due to the torsional effect of the neck. This suggests that the pelvic floor alone is capable of exerting a torque on an elongated object passing through it as the rotating trunk did not contact the bony pelvis while engaging the pelvic floor. The torque is specifically directed to rotate the elongated object so that the longer dimension engages the pelvic floor antero-posteriorly as opposed to a transverse or oblique orientation.

The most important conclusion about the mechanics of the cardinal movements is that the movements seem to be caused by a more complex set of factors than initially theorized. The basic compliance of the bony structures was not sufficient to observe the cardinal movements. The major importance in the occurrence of the cardinal movements was shown to be played by the shape of the maternal pelvis and in particular the ischial spines.

The fetal head shape and size were demonstrated to have a major effect as well. The pelvic floor is also playing a major role by serving as a pivot for

the internal rotation, contributing to the extension and lastly to the external rotation (restitution) of the head. Apart from these major factors, the presence of ligaments, the fetal mandible, initial orientation and other factors were shown to have additional positive effects.

The experiments with the varying fetal head size, demonstrated a clear correlation between the duration of labour and the size of the fetal head. The only exception was observed when the fetal head reduced in size considerably, which undertook a completely different trajectory as compared to the previous simulations. The head rotated internally in the opposite direction and was born in the occiput posterior (OP) orientation. This delivery in this presentation required more effort than the bigger head sizes.

A simulation of brow presentation was also performed. The brow presentation lead to the delivery only after the extended head returned to its normally flexed orientation. This finding further suggests that the internal rotation is dependent on the successful flexion. However, further experiments with an over-flexed head showed issues, suggesting that only a normal flexion level is optimal for the occurrence of full internal rotation.

Applications The presented BirthView simulation system in its current state could already be tested in a real training scenario. Given that further minor optimizations and integrations are completed, the system could be integrated with a haptic feedback device and be used as part of the education of midwives and obstetricains. The trainees could observe the simulated process, while being able to interactively affect it and observe the outcomes. The general purpose haptic devices like the Phantom Omni [1] could be used to simulate the vacuum extraction cup placement scenarios. A more specialized haptic device like the BirthSIM system (Silveira et al., 2004) could be integrated with the simulation system presented in this thesis for a more holistic approach to obstetrical training. The system presented by Silveira et al. (2004) could be extended to allow rotational motions of the fetal head to show cardinal movements. The trainees then could interact with the system to learn the techniques for assisting birth in cases when some of the cardinal movements fail to occur.

[1] http://www.geomagic.com/en/products/phantom-omni/overview

The main goal for the future development of the childbirth simulation system is to achieve reliable predictive capabilities. In its current state it is lacking patient specificity, as the meshes and material models are based on references and literature. Given that patient specific meshes and material properties are available, the system could be used to predict patient specific outcomes.

Much more additional steps are required, but with the future research directions outlined in this thesis it should be possible to start to achieve these predictive capabilities.

6.2 Summary

A real-time childbirth simulation has been developed based on the TLED method (Miller et al., 2007). A projection based contact method has been introduced based on the one presented by Yastrebov (2013). An approach for contact force calculation was presented that combines the benefits of the projection based contact method with TLED. The TLED method implementation with GPU acceleration was also presented with a more general description of the software framework. Further, a number of experiments were performed for validation. The validation experiments showed small errors when compared to commercial FE software, while demonstrating considerably higher computation speed. The next set of experiments investigated the potential of the simulation system to exhibit the cardinal movements of the second stage of labour. These experiments successfully showed the ability to consistently demonstrate the occurrence of the cardinal movements. The presented simulation is created in a fully forward-engineered manner with no imposed trajectories. Furthermore, the causes of the observed cardinal movements were investigated and findings presented and clarified wherever possible

6.3 Limitations and Future work

6.3.1 Limitations

The BirthView simulation system has a number of limitations. The limitations are mostly due to time constraints and the lack of readily available data. The list of the BirthView system's limitations includes:

1. Low robustness of TLED and contact model to highly dynamic processes
2. Lack of accuracy in terms of simulation time as simulated labour proceeds faster than real labour
3. Limited validation for simulation outcomes due to the lack of data

The potential ways of overcoming these limitations are given below. These are not necessarily exhaustive and alternative approaches are possible.

The current performance capabilities of our TLED solver were discussed in Section 5.2.3. They were shown to be sufficient to perform the simulations of complexity such as described in this thesis. However, having higher quality meshes could be of benefit when attempting predictive applications. A detailed patient specific model could require high number of elements in the mesh.

Additionally, faster solver would allow having smaller timesteps while maintaining real-time performance. The decreased timesteps would result in a more robust simulation allowing for more dynamic simulations.

More performance and stability can be gained if the ROM FEM is implemented as part of the simulation system. The approach outlined by Taylor et al. (2010) and Taylor et al. (2011) demonstrate that ROM can allow for larger timesteps while retaining good accuracy.

When patient specific geometry and materials can be extracted, it would become possible to perform simulations of real childbirth cases and then perform validation using dynamic MRI of the birth (assuming the birth was recorded). The comparison with the scan recording and the simulated outcome would be a good way to validate the simulation.

6.3.2 Patient specific mesh for maternal and fetal anatomy

The ultimate goal is to develop a patient specific simulation system capable of predicting potential outcomes of various real deliveries. One of the most important steps required for a patient specific simulator is the ability to quickly and accurately segment the necessary patient specific maternal and fetal structures from MRI (fetus and mother) and ultra-sound (fetus) data. The implementation of a supervised or automatic segmentation system would help. The segmentation could be fully automated or supervised depending on the accuracy of the automatic segmentation methods employed. The segmented models could be pre-processed and imported into the simulation environment automatically. The resulting simulation would become patient specific at least from a anatomical perspective.

6.3.3 Patient specific material properties

The material properties can vary considerably from patient to patient. The age of the mother should be taken into consideration as the prime factor in determining the material properties of the pelvic floor. However, the relationship between the age and the material properties needs to be established reliably and accurately first. Research investigating this relationship is reported in Sec. 2 but not enough data is available to build a reliable predictive model. Therefore, we propose that a detailed study should be conducted to build such a predictive model which would allow the childbirth simulator to adopt the material properties of the maternal structures based on the patient-specific information of the pregnant woman.

6.3.4 Full uterus model with active muscle model

The simple model currently used to simulate the maternal uterine expulsion force could be improved. The model used by Buttin et al. (2009) is more appropriate, but could be further improved upon.

6.3.5 Full cervix model with full physical model of dilation

One of the limitations of the presented BirthView simulation system is that the FE model of the cervix is only represented by a Neo-Hookean hyper-elastic constitutive model. No mechanisms that allow the restructuring of the fibrous matter in the cervical area were implemented. This allows us to simulate passive behaviour of the cervix when the fetal head progresses through it, but will not simulate the process of effacement and dilation. The presence of a valid cervical dilation model would allow simulating the first stage of labour where more detailed observations of the fetal head moulding can be made. The research by Rice et al. (1976) presents a method for simulating the process of effacement and dilation. Lapeer's (Lapeer and Prager, 2001) research on moulding could be integrated into the simulation as well to improve realism.

6.3.6 Deformable fetal head model

The presence of a realistic deformable fetal head model would allow simulating the phenomenon of fetal head moulding during the first and the second stages of labour. The model presented by Lapeer and Prager (2001) is a suitable candidate for integration, which already provides the geometry and the material properties required for integration into the BirthView simulation.

An issue with integration with the current simulation system lies in the fact the GPU TLED solver will have major difficulties in simulating the stiffer parts of the fetal skull, with stiffness moduli of the order of GPa. For details of why a FE model of the bony skull could not be incorporated into the simulation refer to Section 3.2.5. This means an implicit FE method would be more suitable resulting in a hybrid explicit/implicit FE based simulation.

6.3.7 Deformable fetal body with a realistic shoulder model

The current fetal trunk model is rigid and is unable to realistically adapt to the birth canal. The fetal extension in our simulation is limited as the fetal shoulders should also move along with the head. This does not occur as the trunk is rigid.

The mechanisms of the shoulder dystocia could also be investigated if a real-

istic model of the fetal shoulders was incorporated in the simulation system. The anterior shoulder can get stuck hear the maternal pubic symphisis and have adverse effects on the labour outcome. Being able to simulate or potentially predict such scenario would be of great benefit.

6.3.8 Active and anisotropic pelvic floor muscle models

The pelvic floor muscles demonstrate anisotropic behaviour. While this may have little effect on the occurrence of the cardinal movements, it should be noted that introducing the anisotropic behaviour will improve accuracy and could be important for patient-specific predictive applications. The results reported by Li et al. (2011) indicate that the anisotropic behaviour of the muscles has effect on the childbirth.

Bibliography

Aishah, A. (2011). *The association between fetal position at the onset of labour and birth outcomes*. PhD thesis, University of Birmingham Research Archive. 234

Allard, J., Cotin, S., Faure, F., Bensoussan, P.-J., Poyer, F., Duriez, C., Delingette, H., and Grisoni, L. (2007). Sofa-an open source framework for medical simulation. In *MMVR 15-Medicine Meets Virtual Reality*, volume 125, pages 13–18. IOP Press. 47

Apple Inc (2013). *OpenCL Programming Guide for Mac*. Apple Inc. 152

Arulkumaran, S. (2016). *Best Practice in Labour and Delivery*. Cambridge University Press. 61

Ashton-Miller, J. A. and DeLancey, J. O. (2009). On the Biomechanics of Vaginal Birth and Common Sequelae. *Annual Review of Biomedical Engineering*, 11(1):163–176. 32, 36, 40, 56, 197, 233

Audinis, V. (2017). *Computer-based simulation of the effects of instrumental delivery on the fetal head*. PhD thesis, Computing Science. 22

Badir, S., Bajka, M., and Mazza, E. (2013). A novel procedure for the mechanical characterization of the uterine cervix during pregnancy. *Journal of the Mechanical Behavior of Biomedical Materials*, 27:143 – 153. 35

Bamberg, C., Rademacher, G., Güttler, F., Teichgräber, U., Cremer, M., Bührer, C., Spies, C., Hinkson, L., Henrich, W., Kalache, K. D., et al. (2012). Human birth observed in real-time open magnetic resonance imaging. *American journal of obstetrics and gynecology*, 206(6):505–e1. 20

Bathe, K.-J. (1995). *Finite Element Procedures*. Prentice Hall, 1st edition. 44, 86, 90, 154

Belytschko, T., Liu, W., Moran, B., and Elkhodary, K. (2014). *Nonlinear Finite Elements for Continua and Structures*. Wiley. 75, 76, 78, 79, 82, 83, 86

Blender Online Community (2015). *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam. 68

Borell, U. and Fernstrom, I. (1957). The movements in the mechanism of disengagement with special reference to the attitude of the foetal head. *Acta Obstetricia et Gynecologica Scandinavica*, 36(3):347–355. 17, 18, 20

Borell, U. and Fernstrom, I. (1958). Radiographic studies of the rotation of the fœtal shoulders during labour. *Acta Obstetricia et Gynecologica Scandinavica*, 37(1):54–61. 3, 17, 18, 205, 245

Borell, U. and Fernström, I. (1959). Internal anterior rotation of the fœtal head a contribution to its explanation. *Acta obstetricia et gynecologica Scandinavica*, 38(1):103–108. 17, 18

Brandão, F. S. Q. D. S., Parente, M. P. L., Rocha, P. A. G. G., Saraiva, M. T. D. Q. E. C. D. M., Ramos, I. M. A. P., and Natal Jorge, R. M. (2015). Modeling the contraction of the pelvic floor muscles. *Computer Methods in Biomechanics and Biomedical Engineering*, 5842(July):1–10. 27

Bro-Nielsen, M. (1997). Fast finite elements for surgery simulation. *Studies in health technology and informatics*, 39:395–400. 45

Bro-Nielsen, M. and Cotin, S. (1996). Real-time volumetric deformable models for surgery simulation using finite elements and condensation. *Computer Graphics Forum*, 15(3):57–66. 44

Buhimschi, C. S., Buhimschi, I. A., Malinow, A. M., Saade, G. R., Garfield, R. E., and Weiner, C. P. (2003). The forces of labour. *Fetal and Maternal Medicine Review*, 14(4):273–307. 3, 4

Buttin, R., Zara, F., Shariat, B., and Redarce, T. (2009). A biomechanical model of the female reproductive system and the fetus for the realization of a childbirth virtual simulator. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 5263 –5266. v, 29, 37, 40, 41, 42, 56, 197, 249

Caldwell, W. E. and Moloy, H. C. (1938). Anatomical Variations in the Female Pelvis: Their Classification and Obstetrical Significance: (Section of Obstetrics and Gynaecology). *Proceedings of the Royal Society of Medicine*, 32(1):1–30. 5

Cirak, F. and West, M. (2005). Decomposition contact response (DCR) for explicit finite element dynamics. *International Journal for Numerical Methods in Engineering*, 64(8):1078–1110. 54, 120

Comas, O., Taylor, Z., Allard, J., Ourselin, S., Cotin, S., and Passenger, J. (2008). Efficient nonlinear fem for soft tissue modelling and its gpu implementation within the open source framework sofa. *Biomedical Simulation*, pages 28–39. 47

Cotin, S., Delingette, H., and Ayache, N. (1999). Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions on Visualization and Computer Graphics*, 5(1):62–73. 45

Dalstra, M., Huiskes, R., Odgaard, A., and van Erning, L. (1993). Mechanical and textural properties of pelvic trabecular bone. *Journal of Biomechanics*, 26(4):523 – 535. 194

d'Aulignac, D., Martins, J., Pires, E., Mascarenhas, T., and Jorge, R. N. (2005). A shell finite element model of the pelvic floor muscles. *Computer Methods in Biomechanics and Biomedical Engineering*, 8(5):339–347. PMID: 16298856. 25

Dejun Jing, James A. Ashton-Miller, J. O. D. (2013). A subject-specific anisotropic visco-hyperelastic finite element model of the female pelvic floor stress and strain during the second stage of labor. *Journal of Biomechanics*, 29(3):997–1003. v, 16, 27, 29, 33, 36, 43, 56

Downing, K. T., Hoyte, L. P., Warfield, S. K., and Weidner, A. C. (2007). Racial differences in pelvic floor muscle thickness in asymptomatic nulliparas as seen on magnetic resonance imaging–based three-dimensional color thickness mapping. *American Journal of Obstetrics and Gynecology*, 197(6):625.e1–625.e4. 70

Drake, R., Vogl, A. W., and Mitchell, A. W. M. (2014). *Gray's Anatomy for Students*. Churchill Livingstone. 64, 68

Dupuis, O., Moreau, R., Pham, M. T., and Redarce, T. (2009). Assessment of forceps blade orientations during their placement using an instrumented childbirth simulator. *BJOG : an international journal of obstetrics and gynaecology*, 116(2):327–32; discussion 332–3. 40, 56

Eberly, D. H. (2014). *GPGPU Programming for Games and Science*. A. K. Peters, Ltd., Natick, MA, USA, 1st edition. 151

Edgar, C. J. (1916). The mechanisms of labour: some experiments and clinical observations. *American Journal of Obstetrics*, (28):p479. 18, 19

Eichstedt (1859). *Zeugung, Geburts-Mechanismus, etc.* 20

Ericson, C. (2004). *Real-Time Collision Detection (The Morgan Kaufmann Series in Interactive 3D Technology)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 105

Faraci, A., Bello, F., and Darzi, A. (2005). Soft tissue deformation using a nonlinear hierarchical finite element model with real-time online refinement. *Studies in health technology and informatics*, 111:137–144. 45

Field, D. A. (1988). Laplacian smoothing and Delaunay triangulations. *Communications in Applied Numerical Methods*, 4(6):709–712. 66

Francavilla, A. and Zienkiewicz, O. C. (1975). A note on numerical computation of elastic contact problems. *International Journal for Numerical Methods in Engineering*, 9(4):913–924. 94

Franklin, W. R. and Akman, V. (1985). Octree data structures and creation by stacking. In *Proceedings of Graphics Interface 85 on Computer-generated Images: The State of the Art*, pages 159–175, New York, NY, USA. Springer-Verlag New York, Inc. 104

Fraser, D. M. and Cooper, M. A. (2009). *Myles' Textbook for Midwives*. Churchill Livingstone. 3

Gabbe, S., Niebyl, J., Simpson, J., and Anderson, G. (1991). *Obstetrics: normal and problem pregnancies*. Churchill Livingstone. v, 12, 194, 196

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional. 143

Gargantini, I. (1982). Linear octtrees for fast processing of three-dimensional objects. *Computer Graphics and Image Processing*, 20(4):365 – 374. 101

Gaul, R. (2013). Component based engine design. http://www.randygaul.net/2013/05/20/component-based-engine-design/. [Online; accessed 19-Dec-2016]. 140

Geiger, B. (1993). *Three-Dimensional Modeling of Human Organs and Its Application to Diagnosis and Surgical Planning*. PhD thesis, Sophia Antropolis, France. AAI3235306. 19, 36, 42, 56

Gerikhanov, Z., Audinis, V., and Lapeer, R. (2013). Towards a forward engineered simulation of the cardinal movements of human childbirth. In *E-Health and Bioengineering Conference (EHB), 2013*, pages 1–4. 21, 56, 201, 203, 206, 244

Geuzaine, C. and Remacle, J. F. (2009). Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331. 72

Göddeke, D. (2011). *Fast and Accurate Finite-Element Multigrid Solvers for PDE Simulations on GPU Clusters*. Logos Verlag Berlin. 147, 148

Gregory, J. (2009). *Game Engine Architecture*. A K Peters/CRC Press. 140

Hacker, N., Gambone, J., and Hobel, C. (2015). *Hacker and Moore's Essentials of Obstetrics and Gynecology*. Hacker & Moore's. Elsevier - Health Sciences Division. v, 8, 10, 17

Häggström, M. (2014). Medical gallery of mikael häggström 2014. *WikiJournal of Medicine*, 1(2). v, 9

Hallquist, J., Goudreau, G., and Benson, D. (1985). Sliding interfaces with contact-impact in large-scale lagrangian computations. *Computer Methods in Applied Mechanics and Engineering*, 51(1):107 – 137. 49

Hansen, G., Douglass, R., and Zardecki, A. (2005). *Mesh Enhancement: Selected Elliptic Methods, Foundations and Applications*. Imperial College Press. 57

Harris, M., Sengupta, S., and Owens, J. D. (2007). Parallel prefix sum (scan) with CUDA. In Nguyen, H., editor, *GPU Gems 3*, chapter 39, pages 851–876. Addison Wesley. 99

Hart, D. (1912). *Guide to midwifery*. Rebman. 19

Heinstein, M. (1997). *An algorithm for enforcement of contact constraints in quasistatic applications using matrix-free solution algorithms*. 52

Heinstein, M. W., Mello, F. J., Attaway, S. W., and Laursen, T. A. (2000). Contact—impact modeling in explicit transient dynamics. *Computer Methods in Applied Mechanics and Engineering*, 187(3):621 – 640. 52, 53, 118, 119, 184, 244

Henry, M., Stapleton, E., and Edgerly, D. (2011). *EMT Prehospital Care*. EMT Prehospital Care. Mosby JEMS/Elsevier. 18

Hibbit, Karlsson, and Sorensen (2007). *ABAQUS/Standard Analysis User's Manual*. Hibbit, Karlsson, Sorensen Inc., USA. 89, 177

Holzapfel, G. (2000). *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. Wiley. 75, 76, 83, 117, 132, 137

Horton, A., Wittek, A., Joldes, G. R., and Miller, K. (2010). A meshless Total Lagrangian explicit dynamics algorithm for surgical simulation. *International Journal for Numerical Methods in Biomedical Engineering*, 26(8):977–998. 47

Hoyte, L., Damaser, M. S., Warfield, S. K., Chukkapalli, G., Majumdar, A., Choi, D. J., Trivedi, A., and Krysl, P. (2008). Quantity and distribution of levator ani stretch during simulated vaginal childbirth. *American Journal of Obstetrics and Gynecology*, 199(August). 27, 31, 36, 56, 69, 195

Hughes, T. (2012). *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Civil and Mechanical Engineering. Dover Publications. 89

Hughes, T., Taylor, R., and Kanoknukulchai, W. (1977). A finite element method for large displacement contact and impact problems. *Formulations and Computational Algorithms in FE Analysis*, pages 468–495. 95

Humphrey, J. D. and Yin, F. C. (1987). On constitutive relations and finite deformations of passive cardiac tissue: I. A pseudostrain-energy function. *Journal of biomechanical engineering*, 109(4):298–304. 25

Hutton, D. V. (2004). *Fundamentals of Finite Elements*. Elizabeth A. Johnes, New York, 1 edition. 74

Irving, G., Teran, J., and Fedkiw, R. (2006). Tetrahedral and hexahedral invertible finite elements. *Graphical Models*, 68(2):66–89. 46

Jamin, C., Pion, S., and Teillaud, M. (2016). 3D triangulations. In *CGAL User and Reference Manual*. CGAL Editorial Board, 4.9 edition. 71

Janda, S. (2006). *Biomechanics of the pelvic floor musculature*. PhD thesis. 25, 28, 34, 36

Janda, S., van der Helm, F. C., and de Blok, S. B. (2003). Measuring morphological parameters of the pelvic floor for finite element modelling purposes. *Journal of Biomechanics*, 36(6):749 – 757. 25, 31, 33

Jing, D. and Valadez, S. (2011). *Experimental and Theoretical Biomechanical Analyses of the Second Stage of Labor*. BiblioBazaar. 27, 33

Johnsen, S. F., Taylor, Z. A., Clarkson, M., Thompson, S., Hu, M., Gurusamy, K., Davidson, B., Hawkes, D. J., and Ourselin, S. (2012). Explicit contact modeling for surgical computer guidance and simulation. volume 8316. 52

Johnsen, S. F., Taylor, Z. A., Clarkson, M. J., Hipwell, J., Modat, M., Eiben, B., Han, L., Hu, Y., Mertzanidou, T., Hawkes, D. J., and Ourselin, S. (2014). NiftySim: A GPU-based nonlinear finite element package for simulation of soft tissue biomechanics. *International Journal of Computer Assisted Radiology and Surgery*, 10(7):1077–1095. 48, 52, 88

Johnsen, S. F., Taylor, Z. A., Han, L., Hu, Y., Clarkson, M. J., Hawkes, D. J., and Ourselin, S. (2015). Detection and modelling of contacts in explicit finite-element simulation of soft tissue biomechanics. *International Journal of Computer Assisted Radiology and Surgery*, 10(11):1873–1891. 53, 91, 119

Joldes, G. R., Wittek, A., and Miller, K. (2010). Real-Time Nonlinear Finite Element Computations on GPU - Application to Neurosurgical Simulation. *Computer methods in applied mechanics and engineering*, 199(49-52):3305–3314. 46, 52

Kheddar, A., Devine, C., Brunel, M., Duriez, C., and Sibony, O. (2004). Preliminary design of a childbirth simulator haptic feedback. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3270 – 3275 vol.4. 29, 56

Kikuchi, N. and Oden, J. T. (1988). *Contact problems in elasticity: a study of variational inequalities and finite element methods*. SIAM. 49, 50, 118

Krenk, S. (2009). *Non-linear modeling and analysis of solids and structures*. Cambridge University Press. 87

Labelle, F. and Shewchuk, J. R. (2007). Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. *ACM Transactions on Graphics*, 26(3):57. 71

Lapeer, R. (1999). *A biomechanical model of foetal head moulding*. PhD thesis, Cambridge, UK. 66, 67

Lapeer, R., Audinis, V., Gerikhanov, Z., and Dupuis, O. (2014a). A Computer-Based Simulation of Obstetric Forceps Placement. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014*, 8674:57–64. 21, 22

Lapeer, R., Chen, M. S., and Villagrana, J. (2004). An augmented reality based simulation of obstetric forceps delivery. *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, (ISMAR):274–275. 30, 56

Lapeer, R. J., Audinis, V., and Gerikhanov, Z. (2014b). Simulation of vacuum extraction during childbirth using finite element analysis. In *SIMULIA Regional User Meeting (RUM 2014)*. 22

Lapeer, R. J. and Prager, R. W. (2001). Fetal head moulding: finite element analysis of a fetal skull subjected to uterine pressures during the first stage of labour. *Journal of Biomechanics*, 34(9):1125–1133. vi, 24, 25, 30, 59, 60, 194, 228, 250

Larsson, T. and Akenine-Möller, T. (2006). A dynamic bounding volume hierarchy for generalized collision detection. *Comput. Graph.*, 30(3):450–459. 115

Lawlor, O. S. (2003). Embedding OpenCL in C++ for Expressive GPU Programming. *First International Workshop on Domain Specific Languages and High Level Frameworks for High Performance Computing WOLFHPC 2011*. 150

Lee, S.-L., Darzi, A., and Yang, G.-Z. (2005). Subject specific finite element modelling of the levator ani. *MICCAI*, 8(Pt 1):360–367. 26

Lepage, J., Jayyosi, C., Lecomte-Grosbras, P., Brieu, M., Duriez, C., Cosson, M., and Rubod, C. (2015). Biomechanical pregnant pelvic system model and numerical simulation of childbirth: impact of delivery on the uterosacral ligaments, preliminary results. *International urogynecology journal*, 26(4):497–504. 29, 34, 35, 36, 42, 43, 44, 56, 61, 195

Lewy H., Friedrichs K., C. R. (1928). Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische Annalen*, 100:32–74. 89

Li, X., Kruger, J. A., Nash, M. P., and Nielsen, P. M. (2010). Effects of fetal head motion on pelvic floor mechanics. In *Computational Biomechanics for Medicine*, pages 129–137. Springer Nature. 27, 29, 34, 35, 37, 43, 56, 69

Li, X., Kruger, J. A., Nash, M. P., and Nielsen, P. M. F. (2011). Anisotropic effects of the levator ani muscle during childbirth. *Biomechanics and Modeling in Mechanobiology*, 10(4):485–494. 33, 56, 251

Liao, J., Buhimschi, C., and Norwitz, E. (2005). Normal labor: mechanism and duration. *Obstet Gynecol Clin North Am*, 32(2):145–64, vii. 12

Lien, K.-C., DeLancey, J. O. L., and Ashton-Miller, J. A. (2009). Biomechanical analyses of the efficacy of patterns of maternal effort on second-stage progress. *Obstetrics & Gynecology*, 113(4):873–880. 31, 35, 69

Lien, K.-C., Mooney, B., DeLancey, J. O. L., and Ashton-Miller, J. a. (2004). Levator ani muscle stretch induced by simulated vaginal birth. *Obstetrics and gynecology*, 103(1):31–40. 30, 31, 36, 56, 69

Lorensen, W. E., Cline, H. E., Lorensen, W. E., and Cline, H. E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques - SIGGRAPH '87*, volume 21, pages 163–169, New York, New York, USA. ACM Press. 63

Luo, J., Chen, L., Fenner, D. E., Ashton-Miller, J. A., and DeLancey, J. O. L. (2015). A multi-compartment 3-D finite element model of rectocele and its interaction with cystocele. *Journal of biomechanics*, 48(9):1580–6. 28

Marchandise, E., Compere, G., Willemet, M., Bricteux, G., Geuzaine, C., and Remacle, J.-F. (2010). Quality meshing based on stl triangulations for biomedical simulations. *International Journal for Numerical Methods in Biomedical Engineering*, 26(7):876–889. 63

Martins, J. a. a. C., Pato, M. P. M., Pires, E. B., Natal Jorge, R. M., Parente, M., and Mascarenhas, T. (2007). Finite element studies of the deformation of

the pelvic floor. *Annals of the New York Academy of Sciences*, 1101:316–334. 19, 31, 32, 42, 56

McPherson, G. K. and Kriewall, T. J. (1980a). The elastic modulus of fetal cranial bone: A first step towards an understanding of the biomechanics of fetal head molding. *Journal of Biomechanics*, 13(1):9 – 16. 24

McPherson, G. K. and Kriewall, T. J. (1980b). Fetal head molding: An investigation utilizing a finite element model of the fetal parietal bone. *Journal of Biomechanics*, 13(1):17 – 26. 24

Melchert, F., Wischnik, A., and Nalepa, E. (1995). The prevention of mechanical birth trauma by means of computer aided simulation of delivery by means of nuclear magnetic resonance imaging and finite element analysis. *Asia-Oceania Journal of Obstetrics and Gynaecology*, 21(2):195–207. 29, 56

Meyers, S. (2014). *Effective Modern C++: 42 Specific Ways to Improve Your Use of C++11 and C++14*. O'Reilly Media. 143

Miller, K., Joldes, G., Lance, D., and Wittek, A. (2007). Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. *Communications in Numerical Methods in Engineering*, 23:121–134. 46, 47, 48, 52, 78, 79, 81, 163, 247

Miller, T. (1997). A fast triangle-triangle intersection test. *Journal of Graphics Tools*, 2:25–30. 107, 108

Misra, R. (2006). *Ian Donald's Practical Obstetric Problem, 6/e*. B.I. Publications Pvt. Limited. 196

Mitsuhashi, N., Fujieda, K., Tamura, T., Kawamoto, S., Takagi, T., and Okubo, K. (2009). BodyParts3D: 3D structure database for anatomical concepts. *Nucleic Acids Research*, 37(Database):D782–D785. vi, 61, 62, 63, 65, 66, 70

Möller, T. and Trumbore, B. (2005). Fast, minimum storage ray/triangle intersection. In *ACM SIGGRAPH 2005 Courses*, page 7. ACM. 110

Moreau, R., Pham, M., Redarce, T., and Dupuis, O. (2008). Simulation of forceps extraction on a childbirth simulator. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1100 –1105. 29, 40

Moreau, R., Pham, M. T., Silveira, R., Redarce, T., Brun, X., and Dupuis, O. (2007). Design of a new instrumented forceps: Application to safe obstetrical forceps blade placement. *IEEE Transactions on Biomedical Engineering*, 54(7):1280–1290. 29, 40

Morton (1966). A computer oriented geodetic data base and a new technique in file sequencing. Technical Report Ottawa, Ontario, Canada. 103

Müller, M., Dorsey, J., and McMillan, L. (2002). Stable Real-time Deformations. *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 49 – 54. 45

Munshi, A., Gaster, B., Mattson, T. G., Fung, J., and Ginsburg, D. (2011). *OpenCL Programming Guide*. Addison-Wesley Professional, 1 edition. 151, 165

Nagaoka, T., Watanabe, S., Sakurai, K., Kunieda, E., Watanabe, S., Taki, M., and Yamanaka, Y. (2003). Development of realistic high-resolution whole-body voxel models of japanese adult males and females of average height and weight, and application of models to radio-frequency electromagnetic-field dosimetry. *Physics in medicine and biology*, 49(1):1. 61

Noakes, K. F., Pullan, A. J., Bissett, I. P., and Cheng, L. K. (2008). Subject specific finite elasticity simulations of the pelvic floor. *Journal of Biomechanics*, 41(14):3060 – 3065. 26

Noritomi, P., da Silva, J. L., Dellai, R. A., Fiorentino, A., Giorleo, L., and Ceretti, E. (2013). Virtual modeling of a female pelvic floor and hypothesis for simulating biomechanical behavior during natural delivery. *Procedia {CIRP}*, 5:300 – 304. First {CIRP} Conference on BioManufacturing. 36

Oden, J. T. (2010). Finite element method. 5(5):9836. revision 122201. 74

Olovsson, L., Simonsson, K., and Unosson, M. (2005). Selective mass scaling for explicit finite element analyses. *International Journal for Numerical Methods in Engineering*, 63(10):1436–1445. 191, 192

Olshausen (1906). Zur Lehre vom Geburtsmechanismus. *Zentralb.* 20

Paramore, R. H. (1909). A critical inquiry into the causes of the internal rotation of the fœtal head. *BJOG: An International Journal of Obstetrics and Gynaecology*, 16(4):213–232. 17, 18

Parente, M. P., Jorge, R. M. N., Mascarenhas, T., and Silva-Filho, A. L. (2010). The influence of pelvic muscle activation during vaginal delivery. *Obstetrics & Gynecology*, 115(4):804–808. 29, 32, 240

Parente, M. P. L., Jorge, R. M. N., Mascarenhas, T., Fernandes, and Martins, J. a. C. (2008). Deformation of the pelvic floor muscles during a vaginal delivery. *International Urogynecology Journal and Pelvic Floor Dysfunction*, 19:65–71. 32, 42, 56, 69

Parente, M. P. L., Jorge, R. M. N., Mascarenhas, T., Fernandes, a. a., and Martins, J. a. C. (2009). The influence of an occipito-posterior malposition on the biomechanical behavior of the pelvic floor. *European Journal of Obstetrics Gynecology and Reproductive Biology*, 144:166–169. 29, 32, 35, 42, 56, 69

Pebay, P. P. and Baker, T. J. (1991). A comparison of triangle quality measures. In *Proceedings of the National Academy of Sciences*, pages 9653–9657. Plenum Press. 63

Petros, P. (2011). The integral system. *Central European journal of urology*, 64(3):110. 4

Pietrzak, G. and Curnier, A. (1999). Large deformation frictional contact mechanics: continuum formulation and augmented Lagrangian treatment. *Computer Methods in Applied Mechanics and Engineering*, 177(3-4):351–381. 51

Powell, M. J. D. (1969). A method for nonlinear constraints in minimization problems. In Fletcher, R., editor, *Optimization*, pages 283–298. Academic Press, New York. 51

Rao, G. V., Rubod, C., Brieu, M., Bhatnagar, N., and Cosson, M. (2010). Experiments and finite element modelling for the study of prolapse in the pelvic floor system. *Computer Methods in Biomechanics and Biomedical Engineering*, 13(3):349–357. PMID: 20099169. 26

Redon, S., Kheddar, A., and Coquillart, S. (2002). Fast Continuous Collision Detection between Rigid Bodies. *Computer Graphics Forum*, 21(3):279–287. 121

Reinders, J. (2007). *Intel Threading Building Blocks: Outfitting C++ for Multi-core Processor Parallelism*. O'Reilly Media. 145

Rice, D. A., Yang, T. Y., Stanley, P. E., Csapo, A., Danforth, D., Draper, N., Smith, H., Fung, Y., Gillespie, E., Greenhill, J., Holmulund, D., Lindgren, L., , Moloy, H., Romero-Salinas, G., Pantle, J., Aramburú, G., Figueroa, J., Bienarz, J., Caldeyro-Barcia, R., Poseiro, J., Rydberg, E., Scipiades, E., Schwarcz, R., Strada-Saenz, D., Althabe, O., Fernandez-Funes, J., Caldeyro-Barcia, R., Sellheim, H., Mosler, K., Snyder, R., Wahl, F., and Wolf, W. (1976). A simple model of the human cervix during the first stage of labor. *Journal of biomechanics*, 9(3):153–63. 250

Riethmuller, D. and Schaal, J. (2012). *Mécanique et techniques obstétricales*. Sauramps médical. 196

Rubod, C., Brieu, M., Cosson, M., Rivaux, G., Clay, J.-C., de Landsheere, L., and Gabriel, B. (2012). Biomechanical properties of human pelvic organs. *Urology*, 79(4):968.e17 – 968.e22. 27

Rydberg, E. (1935). The significance of the shape of the foetal head in the mechanism of labour. *BJOG: An International Journal of Obstetrics & Gynaecology*, 42(5):795–815. 18, 43

Saleme, C., Parente, M., Jorge, R. N., Pinotti, M., Silva-Filho, A., Roza, T., Mascarenhas, T., and Tavares, J. M. R. (2011). An approach on determining the displacements of the pelvic floor during voluntary contraction using numerical simulation and mri. *Computer Methods in Biomechanics and Biomedical Engineering*, 14(4):365–370. PMID: 21442494. 26

Schöberl, J. (1997). NETGEN An advancing front 2D/3D-mesh generator based on abstract rules. *Computing and Visualization in Science*, 1(1):41–52. 72

Sellheim, H. (1924). Endlich ein echter, weiblicher „kastratoid “. *Arch. Frauenheilk. Konstit.-Forsch*, 10:215–238. 18, 20, 39

Shewchuk, J. R. and Shewchuk, J. R. (2002). Constrained Delaunay Tetrahedralizations and Provably Good Boundary Recovery. *In eleventh International Mesh Roundtable*, pages 193—-204. 71

Shin, D. S., Jang, H. G., Hwang, S. B., Har, D.-H., Moon, Y. L., and Chung, M. S. (2013). Two-dimensional sectioned images and three-dimensional surface models for learning the anatomy of the female pelvis. *Anatomical Sciences Education*, 6(5):316–323. 62

Si, H. and Hang (2015). TetGen, a Delaunay-Based Quality Tetrahedral Mesh Generator. *ACM Transactions on Mathematical Software*, 41(2):1–36. 72, 92

Silveira, R., Pham, M., Redarce, T., Betemps, M., and Dupuis, O. (2004). A new mechanical birth simulator: Birthsim. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3948 – 353. vol.4. 40, 246

Simo, J. and Laursen, T. (1992). An augmented lagrangian treatment of contact problems involving friction. *Computers and Structures*, 42(1):97 – 116. 51

Simo, J. C., Wriggers, P., and Taylor, R. L. (1985). A perturbed lagrangian formulation for the finite element solution of contact problems. *Computer Methods in Applied Mechanics and Engineering*, 50(2):163 – 180. 98

Smyth, C. (1974). Biomechanics and human parturition. *Proceedings of the Royal Society of Medicine*, 67(3):189. 39

Soos, M. (2012). AMD's OpenCL heaven and hell. `https://www.msoos.org/2012/01/amds-opencl-heaven-and-hell/`. 166

Sorbe, B. and Dahlgren, S. (1983). Some important factors in the molding of the fetal head during vaginal delivery - a photographic study. *International Journal of Gynecology & Obstetrics*, 21(3):205–212. 24

Stephenson, R. and O'Connor, L. (2000). *Obstetric and Gynecologic Care in Physical Therapy*. Slack, Incorporated. 17

Stone, J. E., Gohara, D., and Shi, G. (2010a). OpenCL: A parallel programming standard for heterogeneous computing systems. *IEEE Des. Test*, 12(3):66–73. 139, 164, 165

Stone, J. E., Gohara, D., and Shi, G. (2010b). OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems. *Computing in science & engineering*, 12(3):66–72. 150

Strbac, V., Pierce, D., Rodriguez-Vila, B., Vander Sloten, J., and Famaey, N. (2017). Rupture Risk in Abdominal Aortic Aneurysms: A Realistic Assessment of the Explicit GPU Approach. *Journal of Biomechanics*. 48

Strbac, V., Sloten, J. V., and Famaey, N. (2015). Analyzing the potential of GPGPUs for real-time explicit finite element analysis of soft tissue deformation using CUDA. *Finite Elements in Analysis and Design*, 105:79–89. 48

Taylor, L. (1989). *PRONTO 3D: a three-dimensional transient solid dynamics program*. U.S. Dept. of Commerce. 51, 53

Taylor, Z., Comas, O., Cheng, M., Passenger, J., Hawkes, D., Atkinson, D., and Ourselin, S. (2009). On modelling of anisotropic viscoelasticity for soft tissue simulation: Numerical solution and GPU execution. *Medical Image Analysis*, 13(2):234–244. 47

Taylor, Z., Crozier, S., and Ourselin, S. (2010). Real-time surgical simulation using reduced order finite element analysis. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2010*, pages 388–395. 47, 248

Taylor, Z. A., Cheng, M., and Ourselin, S. (2007). Real-Time Nonlinear Finite Element Analysis for Surgical Simulation Using Graphics Processing Units.

In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2007*, pages 701–708. Springer Berlin Heidelberg, Berlin, Heidelberg. 47

Taylor, Z. A., Cheng, M., and Ourselin, S. (2008). High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. *IEEE Transactions on Medical Imaging*, 27(5):650–663. 47, 48, 154, 155, 164

Taylor, Z. A., Crozier, S., and Ourselin, S. (2011). A reduced order explicit dynamic finite element algorithm for surgical simulation. *IEEE transactions on medical imaging*, 30(9):1713–1721. 48, 248

Teran, J., Blemker, S., Hing, V., and Fedkiw, R. (2003). Finite volume methods for the simulation of skeletal muscle. *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, M:68–74. 129

Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. (1987). Elastically deformable models. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques - SIGGRAPH '87*, volume 21, pages 205–214, New York, New York, USA. ACM Press. 44

Tu, C. and Yu, L. (2009). Research on collision detection algorithm based on AABB-OBB bounding volume. In *Education Technology and Computer Science, 2009. ETCS '09. First International Workshop on*, volume 1, pages 331 –333. 115

Varnier (1900). *Obstktrique Journaliere*. Paris. 19, 206, 210

Weisstein, E. W. (2016). Gaussian quadrature. `http://mathworld.wolfram.com/GaussianQuadrature.html`. 82

Wittek, A., Kikinis, R., Warfield, S. K., and Miller, K. (2005). Brain shift computation using a fully nonlinear biomechanical model. *Medical Image Computing and Computer-Assisted Intervention : MICCAI*, 8(Pt 2):583–90. 46

Wittek, A., Miller, K., Kikinis, R., Warfield, S. K., Weisenfeld, N., Mewes, U., Goldberg-Zimring, D., Zou, K., Westin, C.-F., Wells, W., Tempany, C., Golby, A., Black, P., and Jolesz, F. (2007). Patient-specific model of brain deformation:

application to medical image registration. *Journal of biomechanics*, 40(4):919–29. 46

Wriggers, P. (2008). *Nonlinear Finite Element Methods*. Springer Berlin Heidelberg. 49, 50

Yan, X., Kruger, J. A., Nielsen, P. M., and Nash, M. P. (2015). Effects of fetal head shape variation on the second stage of labour. *Journal of Biomechanics*, 48(9):1593–1599. 34, 38, 42, 43, 56

Yastrebov, V. A. (2013). *Numerical Methods in Contact Mechanics*. ISTE/Wiley. 49, 50, 51, 54, 94, 95, 98, 120, 124, 184, 247

Zaki, M. N., Hibbard, J. U., and Kominiarek, M. A. (2013). Contemporary labor patterns and maternal age. *Obstetrics & Gynecology*, 122(5):1018–1024. 61

Zavarise, G. and Lorenzis, L. D. (2009). The node-to-segment algorithm for 2D frictionless contact: Classical formulation and special cases. *Computer Methods in Applied Mechanics and Engineering*, 198(41–44):3428 – 3451. 96