Algorithms for Nonlinear Finite Element-based Modeling of Soft-tissue Deformation and Cutting

ALGORITHMS FOR NONLINEAR FINITE ELEMENT-BASED MODELING OF SOFT-TISSUE DEFORMATION AND CUTTING

BY

BASSMA GHALI, B.Eng.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2008) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Algorithms for Nonlinear Finite Element-based Modeling of Soft-tissue Deformation and Cutting
AUTHOR:	Bassma Ghali B.Eng., (Computer Engineering) Ryerson University, Toronto, Ontario, Canada
SUPERVISOR:	Dr. Shahin Sirouspour
NUMBER OF PAGES:	xiii, 133

To my family

Abstract

Advances in robotics and information technology are leading to the development of virtual reality-based surgical simulators as an alternative to the conventional means of medical training. Modeling and simulation of medical procedures also have numerous applications in pre-operative and intra-operative surgical planning as well as robotic (semi)-autonomous execution of surgical tasks.

Surgical simulation requires modeling of human soft-tissue organs. Soft-tissues exhibit geometrical and material nonlinearities that should be taken into account for realistic modeling of the deformations and interaction forces between the surgical tool and tissues during medical procedures. However, most existing work in the literature, particularly for modeling of cutting, use linear deformation models. In this thesis, modeling of two common surgical tasks, i.e. palpation and cutting, using nonlinear modeling techniques has been studied. The complicated mechanical behavior of soft-tissue deformation is modeled by considering both geometrical and material nonlinearities. Large deformations are modeled by employing a nonlinear strain measure, the Green-Lagrange strain tensor, and a nonlinear stress-strain curve is employed by using an Ogden-based hyperelastic constitutive equation. The incompressible property of soft-tissue material during the deformation is enforced by modifying the strain energy function to include a term

that penalizes changes in the object's area/volume. The problem of simulating the tool-tissue interactions using nonlinear dynamic analysis is formulated within a total Lagrangian framework. The finite element method is utilized to discretize the deformable object model in space and an explicit time integration is employed to solve for the resulting deformations.

In this thesis, the nonlinear finite element analysis with the Ogden-based constitutive equation has also been applied to the modeling of soft-tissue cutting. Element separation and node snapping are used to create a cut in the mesh that is close to the tool trajectory. The external force applied on the object along the tool direction is used as a physical cutting criterion. The possibility of producing degenerated elements by node snapping that can cause numerical instability in the simulation is eliminated by remeshing the local elements when badly shaped elements are generated. The remeshing process involves retriangulation of the local elements using the Delaunay function and/or moving a node depending on what is needed in order to generate elements with the required quality.

Extensive simulations have been carried out in order to evaluate and demonstrate the effectiveness of the proposed modeling techniques and the results are reported in the thesis. A two-dimensional object with a concentrated external force has been considered in the simulations.

Acknowledgements

I, Bassma Ghali, would like to express my gratitude to everyone who contributed to making my master studies a great experience. I wish to single out my supervisor, Dr. Shahin Sirouspour, for his great guidance, support and encouragement during my master studies.

Sincere gratitude to my fellow colleagues in the Haptics, Telerobotics and Computational Vision Lab, Saba Moghimi, Amin Abdossalami, Ramin Mafi, Ivy Zhong, Pawel Malysz, Ali Shahdi, Sina Niakosari, Brian Moody, Behzad Mahdavikhah, Mike Kinsner, Peter Kuchnio, Mahyar Fotoohi and Insu Park for being great friends who made the lab a fun and enjoyable place to study as well as for their help during my graduate program.

I would also like to extent special thanks and appreciations to my family and friends for their continuous support, encouragement and love throughout my live that helped me excel in my academic studies.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Surgical procedures are complex tasks that require a considerable amount of training during residency before doctors can start performing actual surgeries. In the traditional model of training, the resident acquires the skills for performing such complex tasks through attending as many operations as possible in order to learn and participate more in the operation each time. Therefore, the resident would learn by visual cues, repetition and instructive presentations. However, the limited amount of time available for in-hospital resident training and the high financial cost of teaching surgical residents in the operating room suggest the need for an alternative method of training such as virtual surgery training [2, 3] in order to provide better trained surgeons. Virtual surgery training is particularly needed for surgical procedures that are more difficult to learn and require extensive practice. Examples include laparoscopic surgery in which eye-hand coordination is

non-intuitive and neurosurgical procedures which involve delicate tissue manipulations that require a great degree of skill and precision to accomplish the task while avoiding tissue damage.

Recent advances in robotics and information technologies have created new opportunities in the field of medical training. Virtual Reality (VR)-based simulators are emerging as a promising alternative to the conventional means of training. They allow surgeons to practice on virtual patients as they would operate on real patients with realistic sensory feedback. With these simulators, surgical residents could spend time practicing in a virtual environment repetitively with non-restricted access to virtual patient and without any risk. The flexibility offered by the simulator to adjust the tissue properties and operation scenarios without being concerned about patient safety issues is critical for the training process. They can also provide training with different complexity levels and define different measures of performance such as tracking errors and maximum exerted force. Active guidance can be provided by these simulators in the form of corrective force-feedback by the system that would guide the trainee through the task. It can also be provided in the multi user setting of these simulators through motion supervision and corrective forces. In summary, VR training can improve surgical skills in the operating room which helps surgeons make fewer errors and have shorter operative times when performing a surgery after enough training [2–4].

In addition to teaching medical students, surgical simulators would allow more experienced surgeons to keep up with newer surgical techniques and developments. Laparoscopic surgeries is one of these techniques and it will be seen in the

next chapter that many simulators have been developed for training on the essential skills needed to perform such procedures. LaparoscopyVR^{TM} Virtual Reality System shown in Fig. 1.1 is an example of these laparoscopic surgical simulators. Surgeons could also use the simulator as a tool to practice performing surgeries using surgical robotics [5]. Robotic surgeries require additional and special training because they are different from the conventional surgery.

Figure 1.1: LapVR^{TM} virtual reality surgical simulator. "Reproduces by permission of Immersion Corporation. Copyright ©2008 Immersion Corporation. All rights reserved."

These VR-based simulators could also be used as an intra-operative and/or pre-operative planning tool as in [6]. In some procedures, it might be difficult to predict the consequences of some actions or there could be a number of actions among which the best possible must be chosen. Therefore, during a surgery, the surgeon can use the simulator as a strategic planning system to decide the

surgical approach that should be taken. Also, pre-operative planning can be performed (e.g. in percutaneous therapy) where the model is used to plan the operation before the surgery. Patient-specific models can be employed to try different approaches on the virtual model of the patient and then performing the best approach during the surgery. Real-time simulation for assisting robots to perform (semi)-autonomous surgical tasks is another application for surgical simulators.

Robotic surgery is another area that is emerging as an alternative to traditional surgery. These new technologies provide many advantages for doctors and patients at the same time [7–9]. Some of these advantages include enhanced operation precision, increased success rate, reduced trauma to the body, reduced blood loss, less post-operative pain and less risk of infection [9, 10]. It was also found that patients who have a robotic surgery tend to have a shorter hospital stay because of faster recovery which indicates a reduced post-operative health services and cost. Examples of a robotic surgery system are the da Vinci® Surgical System from Intuitive Surgical Inc. shown in Fig. 1.2 and the Zeus® Surgical System from Computer Motion Inc. A simulator that can model the tissue-tool interactions during the operation can be helpful in evaluating the hardware design in the initial development of the controllers without any concerns about safety.

A general surgical simulator can consist of a haptic interface to provide force feedback to the user, a computer system to perform computations and updates on the virtual object that is being simulated and a monitor to provide graphical feedback to the user as it is shown in Fig. 1.3. The user interacts with the virtual model of the organ using the haptic device that functions as surgical tool by specifying the position of the tool. Depending on the tool position, the virtual model

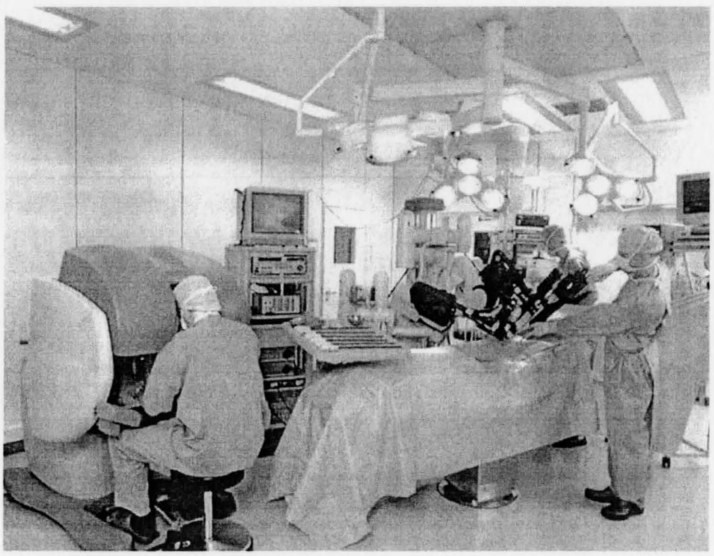


Figure 1.2: The da Vinci® Robotic Surgery System. ©[2008] Intuitive Surgical, Inc.

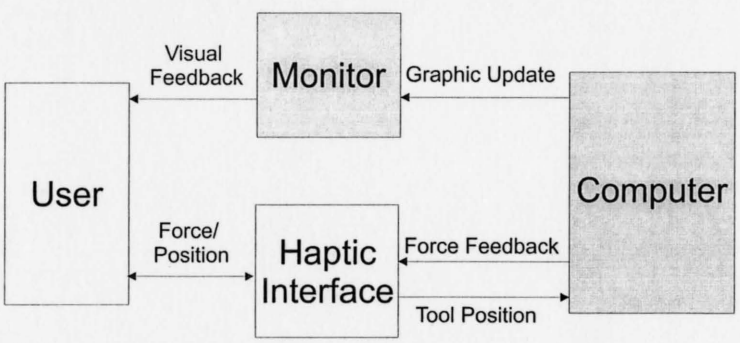


Figure 1.3: Block diagram of a general surgical simulator.

gets updated to provide realistic tool-tissue interaction. The graphical update is presented to the user through the graphical display and the force feedback is provided through the haptic interface.

Surgical procedures involve interactions with rigid and deformable organs like bones and soft-tissues, respectively. Modeling tool-soft tissue interactions (deformations and cutting) is generally much more challenging than modeling tool-bone interactions. For a surgical simulator that involves interactions with a soft organ, a model of soft-tissues that can provide a realistic graphical and haptic feedback to the user is needed. The accuracy is also critical in case of pre-operative task planning and robotic execution of the task. Since soft human ogran's such as brain, liver and kidney are found to exhibit a nonlinear mechanical behavior, a model that takes these complex properties into account should be used. However, most of the time, many simplifications are assumed and a simple model that does not provide the realistic behavior needed is used.

In order to find the displacements and forces in the simulated object during interactions with a surgical tool, the model of the object is discretized in the space and time domains resulting in a set of algebraic equations in terms of the unknown displacements and forces. Many techniques are available for discretizing the object and solving the system of equations. Spatial discretization techniques can provide different levels of accuracy and speed in the simulation depending on the theory they are based on. Continuum mechanics-based methods are usually utilized to model tool-tissue interactions since, in general, they provide a better accuracy.

Finite Element Method (FEM) is an example of a continuum mechanics-based approach that is commonly used since it provides results with better fidelity compared to other techniques. Linear models that have been dominantly applied in the literature to model tool-soft tissue interactions, especially cutting, do not account for the nonlinear mechanical properties of soft-tissues [11, 12]. Therefore, nonlinear models should be employed in order to increase the fidelity in simulating soft-tissue deformations and cutting.

1.2 Problem Statement and Thesis Contributions

This thesis is concerned with the nonlinear modeling of tool-soft tissue interactions for common but important surgical procedures. The objective of this thesis is not to build a simulator for a specific task or tissue; instead, the objective is to use nonlinear models that can potentially improve the accuracy in modeling basic surgical tasks. Since palpation and cutting are among the most common surgical tasks, modeling of these two interactions is investigated in the thesis. Without loss of generality, a two-dimensional object that represents a slice of a three-dimensional soft organ is assumed in the simulations. The modeling involves deforming the tissue after tool-tissue contact, forming a cut when needed, and calculating a force feedback sensation. The contributions of the thesis can be summarized as follows:

- **Problem 1**: Linear elastic FE models, which have been predominantly used in prior relevant work, are inadequate for realistic modeling of soft-tissue deformations and interaction forces with surgical tools. In fact, these models are only valid for small deformations and are incapable of capturing dominant

time-dependent nonlinear behavior of soft-tissues observed in surgery.

Contribution: The Ogden-based nonlinear hyperelastic model presented in [13], which is found to be the best model to represent the soft-tissue behavior observed in experiments, is used as an energy function of the FE model to calculate soft-tissue deformations. Since soft-tissues are considered rubber like material with an almost incompressible response, a term is added to the energy function in order to enforce this condition. The term adds a constrain on the energy function to keep the area/volume constant during deformation by using a relatively large bulk modulus as a Lagrange multiplier and a term that penalizes changes in area/volume. The total Lagrangian formulation is used to formulate the problem and the FEM is employed to spatially discretize the simulated object. An explicit time integration is utilized to discretize the dynamic equations of motion in time for the analysis of the system as in [14] with the addition of damping to the system of equations.

- **Problem 2**: Modeling soft-tissue cutting, which is a fundamental requirement of a surgical simulator, have mostly been done with the assumption of a simple linear elastic behavior of soft-tissues in the literature in order to simplify the analysis as it will be shown in Chapter 2.

  Contribution: The nonlinear FE model that is mentioned in the contribution of Problem 1, is employed to model soft-tissue cutting.

- **Problem 3**: Cutting is a complicated procedure to model as it involves topological modification in the FE mesh. These changes could cause an increase in the number of elements or, more critically, numerical instability depending

on the technique that is used to implement the cut in the FE mesh. After an extensive research on existing cutting modeling techniques in the literature, it is found that node snapping and elements separation in the cutting path is the best method to model cutting since it does not increase the number of elements and it creates a cut that is closest to the cut path. However, this method has the disadvantage of possibly creating degenerated elements that cause numerical instability of the simulation.

Contribution: An algorithm is developed in order to place the snapped node in a position that will guarantee stability and generate a cut path that is closest to the tool trajectory.

The proposed techniques for modeling soft-tissue deformation and cutting are evaluated through simulations that are conducted to demonstrate the performance of the developed algorithms. Many cases that illustrate important aspects of the algorithms are shown.

1.3 Organization of the Thesis

The rest of this thesis is organized as follows. Relevant literature pertaining to surgical simulation systems and modeling of soft-tissue deformation and cutting is presented in Chapter 2. The linear FE modeling process including the formulation of the FEM for linear analysis and the explicit numerical integration method are introduced in Chapter 3. Chapters 4 discusses the nonlinear FE modeling which

covers the total Lagrangian formulation, the deformation gradient, large deformation stress and strain measures along with the hyperelastic constitutive model employed. The algorithm developed for modeling of soft-tissue cutting is presented in Chapter 5. The results of numerical experiments using the proposed nonlinear modeling of deformation and cutting are given in Chapter 6. The thesis is concluded in Chapter 7 where some suggestions for future work are also made.

Chapter 2

Literature Review

This chapter consists of three main sections. The first section is an overview of surgical simulation systems. It provides examples of simulators that have been developed in the past and highlights the importance of haptics in such applications. The main issues that should be considered when modeling soft-tissue deformation are discussed in the second section. These include linear vs. nonlinear mathematical modeling and constitutive equations; different numerical modeling and spacial discretization methods; and how to obtain a numerical solution to the resulting differential equations using time discretization methods. Finally the last section covers issues in the modeling of soft-tissue cutting and surveys different methods that have been previously employed to address these issues.

2.1 Surgical Simulation Systems

A significant amount of research has been devoted to the simulation of different surgical procedures. Surgical simulation systems have many applications and provide surgeons with great advantages as it was explained in Chapter 1. Some of these applications include training, learning new techniques, planning surgeries, executing (semi)-autonomous robotic tasks and designing surgical robots. Many surgical simulators have been developed in the last few years for a wide range of surgical procedures as in [5, 15–20]. In addition, many companies have manufactured surgical simulators for different procedures such as LaparoscopyVR^{TM} Surgical Simulation System by Immersion Medical that was shown in Fig. 1.1, LAP Mentor^{TM} by Simbionix, ProMIS^{TM} surgical simulator by haptica and TempoSurg by VOXEL-MAN. One of the main challenges in the development of these simulators is accurate modeling of tool-tissue interactions which requires the use of complicated mathematical models. This issue will be considered in depth in the rest of this thesis. Another important factor in some surgical simulators is providing the surgeon with a haptic sensation.

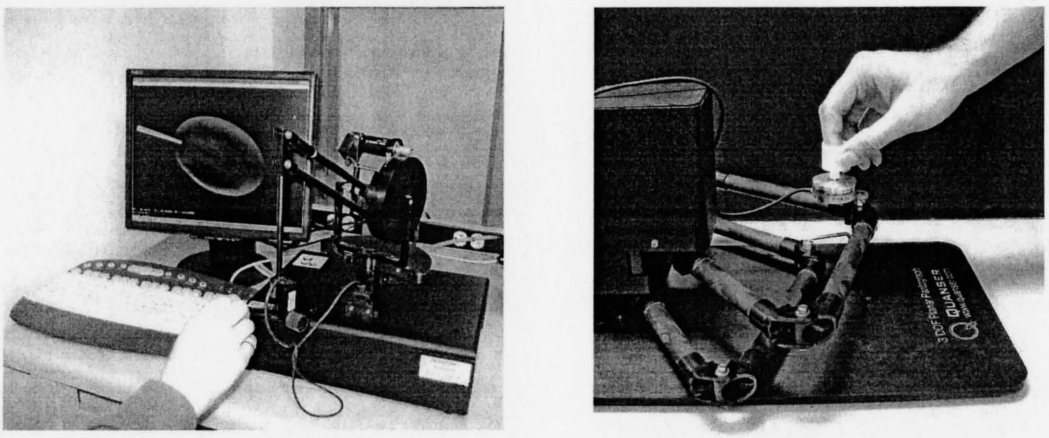
Haptics refers to the sense of touch in virtual environments. It allows human to feel the objects in virtual environments as they are working with real life objects providing a force/kinesthetic feedback to the user through a motorized joystick [21]. One of the main applications for haptics is the medical field and especially in surgery simulation and surgical robots. The sense of force/touch can be very important for the doctor when performing a real surgery using a surgical robot or training using a surgical simulator. Adding force feedback helps the

surgeon or trainer to feel the tool-tissue movement and learn about the mechanical properties of the actual/virtual tissues in contact with the tool [22, 23]. Good mathematical models and force feedback make the performance of the simulator closer to reality which is required to have a better training system.

The importance of haptic in surgical simulation can also be noticed by looking at the literature in the last few years. All the surgical simulation systems mentioned above include force feedback and there has been emphasis on the improvement that these simulators accomplish when the sense of touch is added. Also, most of the work on tissue modeling includes haptics as in [24–27]. The absence of haptic sensation is considered a disadvantage of surgical robots such as da Vinci® and Zeus® [9]. In these systems, the surgeon performs the surgery on the patient only depending on vision cues without feeling the forces applied to the tissues or the texture of the tissues in contact with the surgical tool. This can be very exhausting for the surgeon and can increase the chance of causing errors and tissue damage during surgery because of excessive forces that the surgeon could apply without feeling. Therefore, as it is mentioned in [9], many groups are working on transmitting touch sensation from robotic instruments back to the surgeon.

In the last two-three decades, there has been a lot of research on haptic rendering. Many other applications that use the concept of haptic rendering have been developed over the years. Some of these applications include gaming, pilot training, teleoperation and robotics. In addition, significant research has been done in developing haptic devices that give accurate force feedback to the user some of which are shown in Fig. 2.1.

The direction and magnitude of the force feedback applied by the haptic device



A) B)

Figure 2.1: Examples of haptic devices a) PHANTOM® Premium by SensAble Technologies, Inc. b) 3-DOF Planer Pantograoh System by Quanser Consulting Inc.

depend on the tool position with respect to the virtual object which is specified by the user through moving the haptic interface. The force is computed using the mathematical model employed to represent the virtual object which embodies the geometrical and mechanical properties of the object in the virtual environment. Therefore, for a simulator that involves tool interactions with soft-tissue, a model that represents the mechanical properties of soft-tissues should be used to have a virtual tissue response that is similar to that of human soft-tissues during surgeries. Details regarding different methods to model tool-tissue interactions (deformations and forces) and the issues that should be considered to have accurate deformation modeling are discussed in the next section.

2.2 Modeling of Soft-tissue Deformation

Modeling soft-tissue for computer simulation involves many stages [13]. Since the goal is to simulate soft-tissues during surgeries, it is required to model deformations and internal/external forces in soft-tissues which are mechanical properties that can be analyzed using continuum mechanics. The first stage of the modeling process involves choosing the mathematical model that represents the physical properties and the constitutive equation that best describes the mechanical properties of the soft-tissue. The second stage involves choosing a numerical method to discretize the mathematical model in space in order to be able to solve it numerically using computers. The third stage is solving the discretized equations of motion to calculate the deformation and interaction forces. The following subsections present an overview of different methods that have been used in the literature for each of these stages.

2.2.1 Mathematical Modeling and Constitutive Equations

Linear vs. Nonlinear Modeling of Soft-tissue

Linear analysis of a structural mechanics problem uses an equilibrium equation based on the principal of virtual work in which the displacement field is a linear function of the applied load. This analysis includes the assumption of infinitesimal displacements, i.e., geometrical linearity, and a linear elastic material, i.e., material linearity characterized by a linear stress-strain curve [28, 29]. When these assumptions do not apply to the material that is being modeled, a nonlinear analysis

should be used. Such analysis could involve material nonlinearity that is characterized by a nonlinear stress-strain relation and/or geometrical nonlinearity which includes large displacement, large rotations and large strains.

After a comprehensive review of the literature in bio-mechanical modeling of soft-tissues for surgical simulation, it was found that in many cases these tissues are modeled as a linear elastic material, e.g., see [6,15,17,18,25,26,30]. Linear elastic models are popular due to their computational efficiency which allows for real time simulation. However, linear models are inaccurate in modeling the complex mechanical behavior of human organs' soft-tissues because of their material and geometric nonlinear characteristics. In [11], it is shown that linear elastic models do not model large deformations correctly as they cause distortion in the object being modeled; therefore, geometrical nonlinear model was used with a quadratic strain.

Several methods have been proposed in the literature for simulating complex physical behavior of soft-tissues. In [31], nonlinear FE was used to simulate a deformable body. In [32], a deformable model that is characterized with nonlinear elasticity, material large displacement, anisotropy and incompressibility constraints was simulated using FEM. In [33], soft-tissues such as kidney were modeled as homogeneous, isotropic, incompressible material with nonlinear elasticity behavior. Some work has also been done on modeling brain tissue since it is one of the most complex tissues in the human body. In [12,34], it was found through experiments that mechanical behavior of brain tissue is highly nonlinear with a strong stress-strain rate dependence. Therefore, it was modeled as a single-phase, nonlinear and viscoelastic material. More recent work in [13,35–37] show that, in

addition to the previously mentioned properties, swine brain tissue is considerably softer in extension than in compression. This necessitates the use of a more complicated constitutive model in order to take into account the tissue deformations in tension and in compression as will be discussed later.

Formulations for Nonlinear Analysis

The equilibrium equation of motion for nonlinear analysis can not be solved directly. Therefore, an approximate solution can be found using a previously calculated known equilibrium configuration as a reference for variables and linearizing the resulting equation [29]. When both geometric and material nonlinearities need to be included, which is the case of soft-tissues of human organs, two types of formulations could be employed depending on which equilibrium configuration is used as a reference. The first type is the Total Lagrangian (TL) formulation which is also called Lagrangian formulation. In this approach the original configuration at time 0 is used as reference for all variables. The second type of formulation is called the Updated Lagrangian (UL) formulation in which the last calculated configuration is used as a reference for variables. The choice between these two formulations depends mainly on the numerical effectiveness for the modeling process and on the constitutive law used.

In [14, 31, 38], the TL formulation was used since in this method all variables are referred to the original configuration allowing for the pre-computation of all derivatives with respect to the spatial coordinates. This approach results in fewer mathematical computations compared to the UL formulation in which the derivatives should be recomputed at each time step because the reference configuration

changes. Another difference between the two formulations is the simplicity of the incremental linear strain in UL formulation compared to TL formulation. The initial displacement effect in the incremental linear strain for TL formulation makes the strain-displacement matrix more complex [29].

Constitutive Relations

A constitutive equation is a mathematical model that captures the mechanical properties of the material by describing the relationship between stress and strain. It enters the equilibrium equations of motion in the calculation for stresses which in turn are used to compute the internal forces. The TL and UL formulations mentioned above take into account nonlinear effects caused by large displacements, large rotations and large strains. However, accurate modeling of nonlinear behavior of a specific material depends on the choice of the constitutive equation. There are many constitutive equations available for different types of material in the literature. The reader is referred to [29,39] for a comprehensive review of some of the existing constitutive models.

For elastic materials, which are materials that store energy when loaded, constitutive equations can be used to model linear or nonlinear elasticity depending on the material behavior. In general, constitutive relations are strain energy functions that are used to calculate the stresses in a material by determining their derivative with respect to a strain measure. A constitutive equation consists of constants that should be found to fit experimental data of the material and is based on a measure of deformation. The constitutive equation for linear elastic materials is the simplest form of constitutive equations which assumes a linear relationship between

the stress and strain. In such a model, which is a generalization of the Hooke's law, the stress is a linear function of strain. These equations can not be used to model the nonlinear stress-strain curve and the finite deformations of soft-tissues [13,29]. Therefore, constitutive relations for nonlinear elasticity are needed for this type of materials.

Isotropic hyperelastic materials are nonlinear elastic materials that undergo recoverable large deformations and for which the stress-strain relationship is derived from a strain energy density function such as rubberlike materials [29,39–41]. Soft-tissues of human organs fall into this category of materials because they have the mechanical properties of hyperelastic materials. This type of materials including soft-tissues are generally almost or completely incompressible which means that the area/volume of the tissue does not change during deformations. The incompressibility property is enforced by a constrain on the deformation measure used which indicates that the ratio of a deformed volume to the original volume should be equal to one. Also, the strain energy function can be modified to include this constrain for an almost incompressible material by adding an extra hydrostatic pressure term to the function which is the basis of a deformation/pressure formulation [29]. These constitutive equations are expressed in terms of deformation measures such as strain invariants or principal stretch ratios. Many forms of strain energy functions for isotropic hyperelastic materials are available [29, 39, 41]; however, the most common ones are Ogden [42], Mooney-Rivlin [43], neo-Hookean [44] and Varga [45] models. The Ogden strain energy function has the most general form, whereas the other three functions are special cases of the Ogden function. Other strain energy functions that are not derived

from Ogden function are St. Venant-Kirchhoff [32], Yeoh and Arruda-Boyce functions [39,41].

In [31], Mooney-Rivlin and neo-Hookean were used to model hyperelasticity in order to handle large deformations and nonlinear elasticity that are common in biological tissues. In [33], three constitutive models were employed in order to model the hyperelastic, isotropic and incompressible behavior for kidney and uterus tissues: neo-Hookean, Mooney-Rivlin, and Fung-Demiray. It was found that, for kidney tissues, the Fung-Demiray law with an appropriate choice of parameters produces a better match between numerical and experimental results when compared with the other two constitutive equations. In [32], St. Venant-Kirchhoff elasticity was used as an energy function to model nonlinear elasticity and a constrain was added to the function in order to enforce an anisotropic behavior. It was sound that when St. Venant-Kirchhoff model is employed, modeling the incompressibility property of soft-tissues is difficult and, in most cases, it leads to instability.

In order to model the nonlinear mechanical behavior of brain tissue, in [12,34], a strain energy function that is based on Mooney-Rivlin energy function was employed. The energy was written as a convolution integral with time dependant coefficients to take into account the time-dependent behavior of the brain tissue. However in [35], it was found that the brain tissue has different behavior in compression than in tension which can not be properly modeled using the previous polynomial form of energy function that is based on Mooney-Rivlin. Therefore, another hyperelastic material model based on the Ogden energy function was used

since it is a generalization of the Mooney-Rivlin and neo-Hookean energy functions and it allows the use of fractional power of stretches. Also, this model requires fewer constants to be estimated compared to the second order polynomial hyper-viscoelastic model that was used previously. In [13,35–37], the Ogden-based hyper-viscoelastic model was utilized to simulate brain tissue deformations. It was found that this model accurately represents the tissue behavior in tension and compression for a range of strain rate. Because of the complexity of hyper-viscoelastic model that accounts for most of the computation time, in [13] a simplifying assumption of using a bi-modular Ogden-based hyperelastic equation that is valid for strain rates commonly observed in surgical procedures was proposed. It was demonstrated that the stress calculations of the hyperelastic model are in good agreement with the experimental values for the specified strain rate.

2.2.2 Numerical Modeling and Spacial Discretization Methods

The mathematical model for simulating soft-tissue deformations yields a set of Partial Differential Equations (PDEs) with boundary and initial conditions. In most cases, it is impossible to find an analytical solution to these equations [46]. Therefore, in order to be able to solve such a system of differential equations, the mathematical model should be discretized in space using a numerical method to find a numerical solution. In [22, 23, 47–50] different numerical methods for discretizing deformation models were discussed. These methods could be categorized into two groups based on the approach followed to deform a surface. The first is geometry-based modeling techniques in which deformations are solved by geometrically manipulating the object or the surrounding space. The second is physics-based

modeling techniques in which deformations are solved based on the physics involved in the motion and dynamics of tool-tissue interactions. Preference is given to physics-based techniques over geometry-based techniques when accurate deformation simulation is needed [47,48] because they model the underlying mechanics of the deformation. In addition, in physics-based modeling techniques, the magnitude and direction of forces applied to each node are automatically computed which is needed for haptic rendering. Below is an overview of some of the most commonly used physics-based techniques with their main advantages and disadvantages.

Particle-based Method

In this approach, also known as mass-spring method, the object is represented by a set of point masses (particles) that are connected to each other by a network of springs and dampers. Each particle has its own position in static systems. In dynamic systems, each particle has a velocity and an acceleration component as well. These points move under the effect of internal and external forces applied on the object. Many researchers used this technique to model deformations in soft-tissues as in [19,51,52] because of its simplicity and computational efficiency [22]. However, this method can not be used to model realistic soft-tissue behavior because of the difficulties associated with integrating tissue properties into the particles and constructing an optimal mass-spring network [22,30]. In [53], it was stated that due to the inaccuracy of the mass-spring model, a large number of nodes is required to generate better results which will cause a large system of matrices in addition to

other difficulties that could be encountered such as parameters tuning and modeling incompressibility. Another main drawback of mass-spring systems is that the force computations are strongly related to the topology of the object [23,49]. Therefore, even thought this method is computationally efficient, other methods such as FEM are more accurate in modeling deformations of soft-tissues since they rely on continuum mechanics [23,52].

Boundary Element Method (BEM)

BME is a numerical computation method that can be used to solve PDEs formulated as integral equations [54]. In this method, only the surface of the object is discretized and the given boundary conditions are used to fit boundary values into the integral equations of the boundary nodes in order to solve for the displacements of the boundary nodes. BEM was used in [17,53] to model deformable objects for surgical simulation. It is more computationally efficient than other methods, including FEM, because the interior of the body is not discretized which reduces the size of system of equations that should be solved. However, this can be inaccurate when internal deformations and stresses are important and/or when there is a high surface to volume ratio such as the case with medical applications. Therefore, when internal properties are important or material with complicated nonlinear behavior should be modeled, other volume discretization methods like FEM should be employed [53]. Also, the fully populated matrices generated in BEM cause the storage requirement and computational time to grow by the square of the problem size compared to a linear increase in the FEM using sparse matrices [55].

Finite Element Method (FEM)

In FEM, the object is divided into a mesh of elements that are assembled together through the nodes of these elements in order to find the nodal deformations when a load is applied using interpolation functions over the mesh [29,56,57]. There are many types of elements and shape functions that can be used and their choice depends on the problem being solved and the geometry of the object to be discretized. The properties of each element are formulated into matrices used in the calculations of the unknown nodes displacements and forces. FEM is one of the most popular and most effective methods that have been employed to approximate solutions of PDEs and of integral equations of the continuum mechanics that govern soft-tissue behavior. The majority of the work conducted on modeling deformable objects such as human organs utilized this method, e.g., see [11,13,15,18, 25,27,30,31,58], even though it is generally more computationally expensive than the other methods mentioned earlier.

What makes FEM very popular is the continuum-based approach that needs only few material parameters and the ability to model a multilayered tissues that exhibit complex nonlinear, viscoelastic and anisotropic behavior [22]. A special case is the explicit FEM which is also called tensor-mass approach. In this approach, the physical object is discretized to finite elements just as in the general FEM, but the masses, damping, and forces are lumped on the nodes of the mesh. This method is a transition from the mass-spring method to the FEM and is less computationally expensive than the general FEM. It makes mesh modification in case of cutting easier to achieve because the calculations are done at the element level and there is no need to assemble global matrices [14,49,59].

Meshfree Techniques

Meshfree methods discretize the physical model into a cloud of unconnected points with masses affecting each other according to some mathematical formulation. In [48], a survey of different Lagrangian meshfree methods was presented. It is shown that these methods have been used in many applications including the animation and simulation of deformable bodies. Since these methods avoid using a mesh, they can provide a solution for the drawbacks of FEM such as remeshing which is very computationally intensive when needed during simulation. Also these methods can be utilized for modeling irregular objects where meshing is a complex task. A meshless numerical technique called the Method of Finite Spheres (MFS) was used to model physics-based soft-tissue simulation in [60]. It is a generalization of the FEM that uses the Galerkin formulation to discretize the PDEs which govern the deformable object. However, the displacement field is approximated using functions defined over a spherical area around the node [22]. It was shown that, compared to the solution of FEM, the solution of this method is quite accurate in the area around the tool tip whereas inaccuracy increases away from the tool tip. In addition, in [38], the Element Free Galerkin method (EFG) was used to calculate deformations in soft-tissues.

2.2.3 Numerical Solution and Time Discretization Methods

After Discretizing the system of PDEs of motion for a deformable object using one of the techniques mentioned above, a system of ordinary differential equations is obtained. When a static system that does not take inertia and damping forces into

consideration is being solved, the system of equations is:

$$\mathbf{KU} = \mathbf{R} \tag{2.1}$$

where **K** is the stiffness matrix, **U** is the displacement vector and **R** is a vector of external forces. However, when inertia and damping forces are needed to be taken into account during calculations, which is the case in this thesis, the following dynamic system of equations should be solved:

$$\mathbf{M\ddot{U}} + \mathbf{D\dot{U}} + \mathbf{KU} = \mathbf{R} \tag{2.2}$$

where **M** and **D** are mass and damping matrices; $\ddot{\mathbf{U}}$ and $\dot{\mathbf{U}}$ are acceleration and velocity vectors, respectively. The **K** matrix is constant in the case of a linear elastic system. However, when a nonlinear elastic material is being analyzed, the **K** matrix is not constant and it is dependent on the deformation of the object. In order to solve such a dynamic system of equations, the time-dependent variables have to be discretized in time. Therefore, these equations have to be solved using a time integration scheme in which Eq. 2.2 is satisfied at discrete time points separated by intervals called time steps ($\triangle t$). The position, velocity and acceleration of each node in the discretized object are updated at each simulation time step. The selection of the magnitude of this time step is important because a very large value of time step could cause the result to diverge, while a small value would unnecessarily increase the computations. For dynamic systems, these time integration schemes are divided into two main methods: implicit and explicit. Below is a brief review of each of these schemes. A detailed explanation of these approaches could

be found in [29,30].

Implicit Integration Methods

In implicit time integration methods, the equilibrium conditions at time $t + \Delta t$ are used to calculate the deformations at time $t + \Delta t$. Examples of these methods include Houbolt, Wilson, and Newmark methods [29]. In [18], the trapezoidal rule implicit method was employed to solve the linear dynamic equations of modeling deformations in brain tissues. The main advantages of these methods include their unconditional stability and possibility of using large time steps which could not be used in explicit methods. However, these methods require matrix inversion in order to solve the system of equations and find the deformations of each node in the mesh. This process is computationally expensive when large system of equations is needed to model the object. In some cases, the inverse of the matrix is pre-computed in order to speed up the calculations during time stepping. However, when nonlinear models are used and/or a topological modification occurs which require a change in the stiffness matrix, the pre-computation of the inverse does not help because it has to be recalculated for the new stiffness matrix.

Explicit Integration Methods

In explicit time integration methods, the equilibrium conditions at time t are used to calculate the deformations at time $t + \Delta t$. There are many explicit methods available and they have been employed extensively for simulating models of dynamic deformable objects. In [25], the fourth-order Runge-Kutta was employed for discretizing the time domain instead of the Euler method since it was found

that it can accommodate larger time steps which could lead to a speed-up in the calculations. In [59], three different explicit methods were examined: first-order Euler integration, fixed time step fourth-order Runge-Kutta, and three different formulations of the Verlet algorithms. The Leapfrog Verlet technique was the one chosen because it was found to give the best results compared to the other two methods. In [49], the solution of the governing equations was obtained using a modified-Euler integration method which is an explicit method.

The most common explicit method is the central difference technique which was employed in [11,14,31,38] for computing soft-tissue deformations. In all these publications, an explicit method was selected to avoid solving a system of algebraic equations at every time step which requires setting up and inverting a large sparse matrix or using iterative methods. Also, calculations can be done at the element level when lumped mass and damping matrices are utilized. Therefore, these methods are more computationally efficient than implicit methods. In addition, explicit methods and the central difference method in specific make the treatment of nonlinearities straight forward since they do not require the calculation of the strain incremental stiffness matrices as it is the case for implicit time integration methods. Therefore, the central difference method is the most common explicit method for nonlinear dynamic analysis [14,29]. A main disadvantage of explicit integration schemes is that they are only conditionally stable and their stability depends on the time step chosen for the calculations. The time step has to be smaller than a critical length specified by the material properties and the size of elements in the mesh [14,29,57].

2.3 Modeling Cutting of Soft-tissue

Soft-tissue cutting is one of the common tasks during a surgery. It is also one of the most complicated procedures to model since it involves a topological modification of the underlying model. A literature review on techniques for simulating soft-tissue cutting revealed that most available techniques use linear elastic models to simplify the analysis for a real-time simulation. For example, linear elastic FE models were utilized in [25,59,61] to simulate cutting in deformable objects. In [62], a linear FE mesh was employed as a physical model for a deformable object and a linear mass-spring/particle system was used in the remeshing stage when a topological modification is needed in order to homogenize the mesh in the vicinity of the cut and prevent small elements that cause instability. In [63], a linear mass-spring model was proposed for the modeling of cutting. In [64], a system of point masses that are connected with linear, semi-linear and nonlinear springs/dampers was developed for modeling the cutting. The work presented in [49] utilized a linear elasticity model and a nonlinear strain tensor in order to take into account large deformations of soft-tissues with a linear stress-strain relation.

These linear elastic models can not generate a realistic haptic feedback and deformation of human organs as discussed in the previous section because of the assumptions of a linear stress-strain relationship and infinitesimal deformations embedded in the equations of linear elasticity. In [32, 65], nonlinear elasticity of soft-tissues during deformation and cutting was modeled. However, the energy functions used, such as St. Venant-Kirchhoff and neo-Hookean energy functions, were found to be not the best energy functions to represent the nonlinear behavior in soft-tissues. Therefore, more complex nonlinear formulations that model large

deformations should be employed to improve the realism.

Many mesh cutting techniques have been developed in the literature and a survey of different aspects of these techniques is given in [64]. These techniques in general try to reduce the number of new elements created after a cut, generate new elements with good quality, keep a continuous mesh structure between the elements in the cut path, and create a cut that go through the tool path as close as possible. The main methods that are used to model cutting include removing elements [25,32], subdividing the intersected elements [59,63,64,67,68], separating elements along the cut path [49,61,62,65] and refining followed by separating elements [69].

The element removal technique does not preserve the volume and mass of the model and requires a very fine mesh to generate a smooth cutting path. The element subdivision and refinement followed by separation techniques generate the best fitting cutting path; however, they increase the number of elements in the model after each cut which cause a decrease in the performance of the simulation. The element separation technique does not increase the number of elements in the model, preserves the mass and area of the model and creates a cut that is the closest to the path traversed by the tool using node snapping. The only disadvantage of this method is elements with bad quality (small area or bad shape) could be generated when selected nodes are snapped to create a cut that fits the tool trajectory. This could cause a numerical instability in the simulation since the critical time step decreases and becomes smaller than the utilized time step. The system could continue to be stable if the time step is changed to a smaller value; however, that would slow down the computations. Therefore, if a proper method is employed to

ensure that degenerated elements are not generated when node snapping is performed, element separation is the best method to model the cutting procedure in deformable objects.

Simulating cutting in soft-tissues requires a cutting criterion to determine when the cutting procedure should be performed. This can be a physical criterion, geometrical criterion or both. In [49], geometrical and physical conditions should be met before a cutting procedure was performed. The geometrical criterion determines if the user displacement on the surface of the simulated object corresponds to a cutting attempt or not and the physical criterion checks if the physical interaction between the cutting tool and the object is sufficient to break the object. To this end, the stress in the object that is subjected to an external load by the cutting tool was considered the physical criterion and the object was broken when the maximum stress becomes greater than the material toughness. In [67], if the external force component in the plane of the tool exceeded a tissue dependent threshold force, the scalpel would start cutting. The force at the tool tip was used as a cutting criterion in [66]. When this force reached a threshold value which was obtained from physical experiments, the cut was created in the mesh. In [61–63], the cut surface or line was defined only according to the tool movement.

Since the cutting procedure involves a topological modification and node snapping, badly shaped elements with small area can be generated. These elements are called degenerated elements that can cause numerical instability in the simulation [49, 65]. Therefore, the cutting algorithm should properly handle these cases to avoid instability. In [49, 61], whenever a degenerated element was detected, it was removed with some considerations to minimize the effect on the mass of the

object. However, not all elements can be removed without larger changes to the mesh and such approach can violate the principle of mass conservation. In [62,63], the mesh quality was improved after performing cutting and node snapping by a sequence of local mesh relaxation steps that moves the interior nodes in order to have an even spacing between the nodes in the mesh.

Cutting of soft-tissue can be affected by many factors such as the representation of the tool, the sharpness of the cutting tool, and the speed of applying the load. In [66], liver cutting experiments demonstrated that the critical force for cutting is dependent on the sharpness of the tool tip and the tool velocity. A round tip blade was found to have much higher critical force and a longer initial deformation phase before initiating a cut. It was also determined that the cutting force increases if the tool velocity increases which showed the viscoelastic behavior of soft-tissues. The cut opening displacement was another issue that was examined and found to be dependent on the length and depth of the cut as well as the prestresses in the tissue. The relationship between slicing angle, blade edge geometry, contact length, the fracture force and the applied force were examined through experiments in [70]. In [49], the effect of the tool sharpness was considered by adding a sharpness factor dependent on the size of the tool edge. The calculated stresses in the tool-soft tissue contact area are scaled by this factor to generate higher stresses for sharper tools. Also, the fact that a damage can occur in the cutting area was added by introducing a damage parameter that increases the effect of loading in an already cut area.

Meshfree methods which are numerical techniques that do not use a mesh can

also be utilized to model cutting instead of mesh-based methods. In [22], a mesh-free method was proposed and it was claimed that it is a promising technique to model cutting since it can have the potential of solving some of the problems associated with remeshing when a mesh-based FEM is employed. In [66], a hybrid approach to simulate surgical cutting procedures was proposed. The approach combined a node snapping technique with a physically-based mehsfree computational scheme.

Chapter 3

Linear Finite Element Modeling

In order to simulate a deformable body, a mathematical description of the geometry and elasticity is needed to describe the relation between the displacement introduced by the load and the deformations and forces in the body. The displacement function which assigns a displacement to every point in the deformable body is essentially a mapping with continuous domain and range spaces. The FEM is used in order to discretize the domain of the displacement function. It is the best method to describe the physics of deformable objects such as human organs since it is based on the strong mathematical foundation of continuum mechanics. In the FEM, the body is divided into a finite number of elements and nodes in order to reduce the problem to a finite number of unknowns, in this case nodes displacement. This process is illustrated in Fig. 3.1. After the spatial discretization, the displacement field within each element is described in terms of nodes displacements using an interpolation function.

However, the standard FEM can be computationally expensive and not suitable for topological modifications because that would require modifying the global

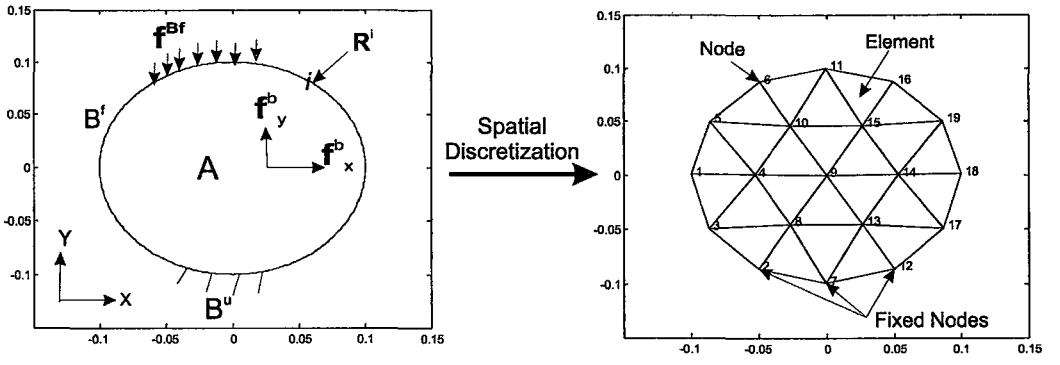


Figure 3.1: A general two-dimensional body and the finite element mesh generated after discretization

matrices of the object. Therefore, an explicit FEM (tensor-mass model) that lumps masses on the nodes have been used in this thesis since it is more computationally efficient than the original FEM and it can be easily adapted to accommodate topological changes due to operations such as cutting.

An explicit integration method, the central difference technique, is utilized to solve the dynamic system of equilibrium equations generated by the finite element method. Such approach can be computationally more efficient than implicit methods since it does not need solving a system of equations. The central difference method is commonly employed for nonlinear dynamic analysis in the literature since it can easily treat nonlinear deformation models. The main disadvantage of this method is that it is only conditionally stable which can occasionally impose a severe restriction on the time step size in the simulation. In particular, a time step that is smaller than the critical time step should be used to guarantee numerical stability. In the case of topological modification where critical time step could change due to the changes in the elements, an algorithm should be developed to ensure that the new elements with the existing simulation time step would not

violate the stability condition.

The first section of this chapter provides an overview of the finite element formulation for linear analysis to show how the method is employed to discretize the equations of motion. It starts by stating the continuous problem that is required to be solved and the principal of virtual work that is the basis of the FEM. A focus on the disretization of the continuum mechanics governing equations using FEM is given next to acquire the discretized equations of motion for a static system. Finally, the derivation of the governing equations for a dynamic system is discussed. In the second section, a description of the central difference method and its application to solve the governing equations of motion is given. The material in this chapter is based on the detailed explanation given in [29] and uses the same notation.

3.1 Formulation of the FEM for Linear Analysis

3.1.1 The principal of Virtual Work

The derivation of the finite element equilibrium equations for a linear elastic material starts with the general elasticity problem that considers the object to be modeled as a two-dimensional linear elastic body A shown in Fig. 3.1. The boundary of the area A consists of two parts: B_u which is the support of the body and is fixed with prescribed displacements $\mathbf{U}^{B_u}$; and B_f a part of the boundary that is subjected to boundary traction $\mathbf{f}^{B_f}$. The area A is subjected to body forces $\mathbf{f}^b$ and an externally applied load force concentrated on finite number of specific points i of the area $\mathbf{R}^i$. All these forces have two components corresponding to the X and

Y coordinate axes.

The body displacement at any internal point on the object is denoted by $\mathbf{U}$ which also has two components, one for each coordinate axis as follows:

$$\mathbf{U}(X,Y) = [U\ V]^T \tag{3.1}$$

The prescribed displacements at the fixed boundary B_u are given by $\mathbf{U} = \mathbf{U}^{B_u}$.

The linear strain field corresponding to the displacement field $\mathbf{U}$ is:

$$\boldsymbol{e} = [e_{xx}\ e_{yy}\ e_{xy}]^T \tag{3.2}$$

where $e_{xx} = \frac{\partial U}{\partial X}$; $e_{yy} = \frac{\partial V}{\partial Y}$; $e_{xy} = \frac{\partial U}{\partial Y} + \frac{\partial V}{\partial X}$

The above relation could be rewritten in a matrix form as $\boldsymbol{e} = \mathbf{B}\,\mathbf{U}$ where $\mathbf{U}$ is defined in 3.1 and $\mathbf{B}$ is the strain-displacement matrix given by:

$$\mathbf{B} = \begin{bmatrix} \frac{\partial}{\partial X} & 0 \\ 0 & \frac{\partial}{\partial Y} \\ \frac{\partial}{\partial Y} & \frac{\partial}{\partial X} \end{bmatrix} \tag{3.3}$$

For linear elastic materials, the stresses corresponding to the strain tensor given in Eq. 3.2 are found through Hooke's law:

$$\boldsymbol{\tau} = \boldsymbol{C}\boldsymbol{e} = [\tau_{xx}\ \tau_{yy}\ \tau_{xy}]^T \tag{3.4}$$

where $\boldsymbol{C}$ is the stress-strain relation matrix.

The objective is to find the deformation field **U** of the body A and the corresponding strains e and stresses τ given the applied loads, boundary conditions and the material stress-strain law. In order to solve this problem, the principal of virtual work can be employed to establish the governing differential equations at the equilibrium, i.e.

$$\int_A e^T \tau \mathrm{d}A = \int_A \mathbf{U}^T \mathbf{f}^b \mathrm{d}A + \int_{B_f} \mathbf{U}^{B_f^T} \mathbf{f}^{B_f} \mathrm{d}l + \sum_i \mathbf{U}^{i^T} \mathbf{R}^i \tag{3.5}$$

Eq. 3.5 states that for any body like the one shown in Fig. 3.1 to be at equilibrium, the total internal virtual work (left hand side of the equation) must be equal to the total external virtual work (right hand side of the equation) when a small virtual displacement applied on the body in its state of equilibrium. It should be noted that the displacement field in Eq. 3.5 is a continuous field and all integrations are performed over the original area without the effect of imposed virtual displacements.

3.1.2 Derivation of FE Equations for Static System

To derive the discretized equations from the continuous equation given in 3.5, the finite element discretization scheme is employed. In this approach, the body shown in Fig 3.1 is divided to discrete triangular finite elements that are interconnected at nodal point. The displacement within each element is assumed to be a function of the nodal point displacement using an interpolation function. Therefore, for each element, the displacement field is:

$$\mathbf{u}_m = \mathbf{H}_m \, \hat{\mathbf{u}} \tag{3.6}$$

where $\mathbf{H}_m \in \mathbb{R}^{2\times 6}$ is the displacement interpolation matrix, the superscript m denotes element m, and $\hat{\mathbf{u}}$ is a vector of element nodal displacement such that:

$$\hat{\mathbf{u}} = [u_1\ v_1\ u_2\ v_2\ u_3\ v_3\ \ldots u_n\ v_n]^T \tag{3.7}$$

and n is the number of nodes per element and the numbers $1, 2, \ldots n$ are the local numbers of the nodes. Based on Eqs. 3.6 and 3.7, for a three node triangular element, the displacement and position of any point within the element can be expressed in terms of the nodal displacements and positions using the interpolation function as follows:

$$u_m = \sum_{n=1}^{3} h_n\ u_n \quad ; \quad v_m = \sum_{n=1}^{3} h_n\ v_n \tag{3.8}$$

$$x_m = \sum_{n=1}^{3} h_n\ x_n \quad ; \quad y_m = \sum_{n=1}^{3} h_n\ y_n \tag{3.9}$$

where h_n is the interpolation function or the isoparametric coordinate of node n. The interpolation functions (h_1, h_2, h_3) are the components of the displacement interpolation matrix $\mathbf{H}_m$ given in Eq. 3.6 and are functions of nodes coordinates as will be detailed in Chapter 6.

Given the discretized deformation field in Eq. 3.7, the continuous strain and stress fields in Eqs. 3.2 and 3.4 can now be written for each element as follows:

$$\boldsymbol{e}_m = \mathbf{B}_m \hat{\mathbf{u}} \tag{3.10}$$

$$\boldsymbol{\tau}_m = \mathbf{C}_m \boldsymbol{e}_m \tag{3.11}$$

where$\mathbf{B}_m \in \mathbb{R}^{3\times 6}$ is the strain-displacement matrix of element m and $\mathbf{C}_m \in \mathbb{R}^{3\times 3}$ is the elasticity matrix of element m.

After the discretization, Eq. 3.5 can be rewritten as a sum of integration over the area of all elements:

$$\sum_{m=1}^{k}\int_{A_m} e_m^T \tau_m \mathrm{d}A_m = \sum_{m=1}^{k}\int_{A_m} \mathbf{u}_m^T \mathbf{f}_m^b \mathrm{d}A_m + \sum_{m=1}^{k}\int_{l_m^1,\ldots,l_m^q} \mathbf{u}_m^{B^T} \mathbf{f}_m^B \mathrm{d}l_m + \sum_i \mathbf{u}^{iT}\mathbf{R}^i \quad (3.12)$$

where k is the total number of elements, $l_m^1,\ldots,l_m^q$ denote the element edges that are part of the boundary of the object and $\mathbf{f}_m^B$ are boundary traction where the superscript B includes the whole boundary.

By substituting Eqs. 3.6, 3.10 and 3.11 into Eq. 3.12, the following equation is obtained:

$$\hat{\mathbf{U}}^T\left[\sum_{m=1}^{k}\int_{A_m}\mathbf{B}_m^T\mathbf{C}_m\mathbf{B}_m\mathrm{d}A_m\right]\hat{\mathbf{U}} = \hat{\mathbf{U}}^T\left[\sum_{m=1}^{k}\int_{A_m}\mathbf{H}_m^T\mathbf{f}_m^b\mathrm{d}A_m + \sum_{m=1}^{k}\int_{l_m^1,\ldots,l_m^q}\mathbf{H}_m^{B\,T}\mathbf{f}_m^B\mathrm{d}l_m + \mathbf{R}_c\right] \quad (3.13)$$

where $\mathbf{R}$ is a vector of the concentrated load forces applied to the nodes of the mesh and $\hat{\mathbf{U}}$ is a vector of global nodal displacement of all nodes in the mesh.

To continue, the following notations are defined based on Eq. 3.13:

$$\mathbf{K} = \sum_{m=1}^{k}\mathbf{K}_m = \sum_{m=1}^{k}\int_{A_m}\mathbf{B}_m^T\mathbf{C}_m\mathbf{B}_m\mathrm{d}A_m \quad (3.14)$$

where K and $\mathbf{K}_m$ are the global and element stiffness matrices, respectively and

$$\begin{aligned}\mathbf{R} &= \mathbf{R}^b + \mathbf{R}^l + \mathbf{R}^c \\ &= \sum_{m=1}^{k} \mathbf{R}_m^b + \sum_{m=1}^{k} \mathbf{R}_m^B + \mathbf{R}^c \\ &= \sum_{m=1}^{k} \int_{A_m} \mathbf{H}_m^T \mathbf{f}_m^b \mathrm{d}A_m + \sum_{m=1}^{k} \int_{l_m^1,\ldots,l_m^q} \mathbf{H}_m^{B\,T} \mathbf{f}_m^B \mathrm{d}l_m + \mathbf{R}^c\end{aligned} \tag{3.15}$$

where $\mathbf{R}$ is the load vector which consists of body forces $\mathbf{R}^b$, boundary forces $\mathbf{R}^B$ and concentrated load forces $\mathbf{R}^c$.

Using Eqs. 3.14 and 3.15, Eq. 3.13 can be rewritten as a system of linear equations in terms of the unknown nodal displacements as follows:

$$\mathbf{K}\,\mathbf{U} = \mathbf{R} \tag{3.16}$$

3.1.3 Derivation of FE Equations for Dynamic System

Eq. 3.16 represents the static equilibrium equation of the assembled mesh in which inertia and damping forces are not taken into account. These forces need to be considered for dynamic simulations in order to model the effect of rapidly applied loads and the energy dissipation during vibration. Such forces can be incorporated into the body force vector $\mathbf{R}^b$ as follows:

$$\mathbf{R}^b = \sum_{m=1}^{k} \int_{A_m} \mathbf{H}_m^T \left[\mathbf{f}_m^b - \rho_m \mathbf{H}_m \ddot{\mathbf{U}} - \kappa_m \mathbf{H}_m \dot{\mathbf{U}}\right] \mathrm{d}A_m \tag{3.17}$$

where $\ddot{\mathbf{U}}$ is a vector of nodes accelerations, ρ_m is the mass density of element m, $\dot{\mathbf{U}}$ is a vector of the nodes velocities and κ_m is the damping property of element m.

By defining the mass matrix as:

$$\mathbf{M} = \sum_{m=1}^{k} \mathbf{M}_m = \sum_{m=1}^{k} \int_{A_m} \rho_m \mathbf{H}_m^T \mathbf{H}_m \mathrm{d}A_m \tag{3.18}$$

and the damping matrix as:

$$\mathbf{D} = \sum_{m=1}^{k} \mathbf{D}_m = \sum_{m=1}^{k} \int_{A_m} \kappa_m \mathbf{H}_m^T \mathbf{H}_m \mathrm{d}A_m \tag{3.19}$$

the equilibrium equation can be rewritten as follows:

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{D}\dot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{R} \tag{3.20}$$

M, D and $\mathbf{K} \in \mathbb{R}^{2N \times 2N}$ and $\ddot{\mathbf{U}}$, $\dot{\mathbf{U}}$ and $\mathbf{U} \in \mathbb{R}^{2N \times 1}$ where N is the number of nodes in a two dimensional mesh. Eq. 3.20 represents the global governing dynamics of a linear-elastic finite-element model for soft-tissue deformation constructed based upon the elemental matrices.These equations must be solved in order to obtain the nodal deformation vector U as a function of time. The assumption of infinitesimal displacements has entered the equilibrium equation in the evaluation of the stiffness matrix **K** and the load vector **R** since all integrations have been performed over the original area of the elements and the strain-displacement matrix is assumed to be constant and independent of elements displacements. The other assumption of linear elastic material has entered by the use of a constant stress-strain matrix **C**.

3.2 Explicit Numerical Integration Method for Solving Dynamic Equations

A time integration method should be employed to solve the discretized equations of motion given in Eq. 3.20 and compute the deformation field **U**. To this end, the time variable is discretized and the deformation variables are only computed at sample times separated by Δt time steps, as shown in Fig. 3.2. Soft-tissue modeling requires an efficient numerical scheme when integrating the equations of motion in the time domain to balance computation speed against numerical accuracy and simulation stability. Therefore, an explicit time integration method, the central difference method is used in this thesis.

In the explicit central difference method, the solution from the previous time step (time=t) is used in order to find the displacement field at the next time step (time=$t + \Delta t$). Therefore, the global system of discretized equations of motion that is used to solve for the displacement at $t + \Delta t$ is:

$$\mathbf{M}\,{}^{t}\ddot{\mathbf{U}} + \mathbf{D}\,{}^{t}\dot{\mathbf{U}} + \mathbf{K}\,{}^{t}\mathbf{U} = {}^{t}\mathbf{R} \tag{3.21}$$

where superscript t means that the solution at time t is considered.

Fig. 3.2 shows the time discretization of the displacement in the central difference method. Difference formulas for approximating the velocity and acceleration in terms of displacements can be derived as follows:

$$Velocity: \qquad {}^{t}\dot{\mathbf{U}} = \frac{1}{2\Delta t}({}^{t+\Delta t}\mathbf{U} - {}^{t-\Delta t}\mathbf{U}) \tag{3.22}$$

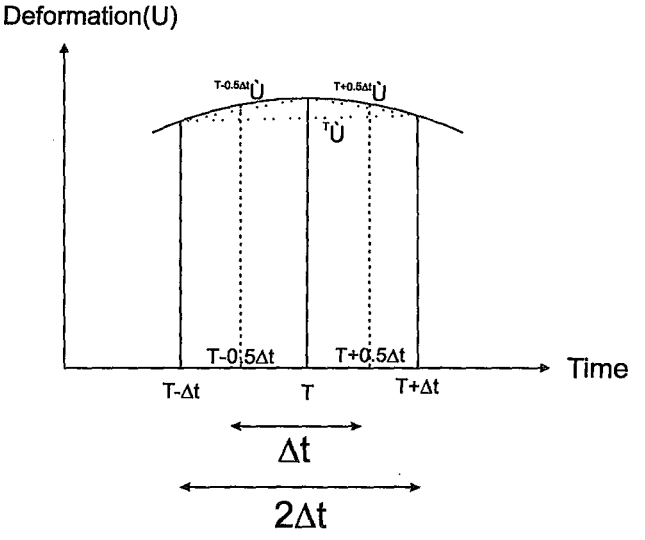


Figure 3.2: Time discretization of the deformation field in the central difference method

$$
\begin{aligned}
\textit{Acceleration}: \qquad {}^{t}\ddot{\mathbf{U}} &= \frac{1}{\Delta t}({}^{t+\frac{1}{2}\Delta t}\dot{\mathbf{U}} - {}^{t-\frac{1}{2}\Delta t}\dot{\mathbf{U}}) \\
&= \frac{1}{\Delta t}\left(\frac{{}^{t+\Delta t}\mathbf{U} - {}^{t}\mathbf{U}}{\Delta t} - \frac{{}^{t}\mathbf{U} - {}^{t-\Delta t}\mathbf{U}}{\Delta t}\right) \\
&= \frac{1}{\Delta t^2}({}^{t+\Delta t}\mathbf{U} - 2{}^{t}\mathbf{U} + {}^{t-\Delta t}\mathbf{U})
\end{aligned}
\tag{3.23}
$$

By substituting Eqs. 3.22 and 3.23 into the system of dynamic equations in 3.21, the following relation is obtained:

$$
\frac{\mathbf{M}}{\Delta t^2}({}^{t+\Delta t}\mathbf{U} - 2{}^{t}\mathbf{U} + {}^{t-\Delta t}\mathbf{U}) + \frac{\mathbf{D}}{2\Delta t}({}^{t+\Delta t}\mathbf{U} - {}^{t-\Delta t}\mathbf{U}) + \mathbf{K}\,{}^{t}\mathbf{U} = {}^{t}\mathbf{R} \tag{3.24}
$$

In order to solve for the deformation field in the next time step at $t+\Delta t$, Eq. 3.24 can be rewritten as follows:

$$
\left(\frac{\mathbf{M}}{\Delta t^2} + \frac{\mathbf{D}}{2\Delta t}\right){}^{t+\Delta t}\mathbf{U} = {}^{t}\mathbf{R} - \left(\mathbf{K} - \frac{2\mathbf{M}}{\Delta t^2}\right){}^{t}\mathbf{U} - \left(\frac{\mathbf{M}}{\Delta t^2} - \frac{\mathbf{D}}{2\Delta t}\right){}^{t-\Delta t}\mathbf{U} \tag{3.25}
$$

To reduce the calculations at each time step, a lumped (diagonal) mass **M** and damping **D** matrices can be employed as suggested in [25, 30]. This technique assumes the damping and mass effects are concentrated at the nodes only and the resulting viscous damping is with respect to ground.

It can be observed from Eq. 3.25 that when an explicit integration method is used in conjunction with diagonal mass and damping matrices, no matrix inversion is needed and calculations can be performed at the element-level. This can be clearly seen in Eq. 3.25 since the force calculation on the right hand side can be done independently for each element, and the left hand side is diagonal. This significantly reduces the computational cost of each time step compared to that of implicit methods. Therefore, there is no need to assemble the stiffness matrix and the computations of the elastic forces are performed for each element separately and contributions of each element are summed to generate the elastic force vector as follows:

$$\mathbf{K}\,{}^{t}\mathbf{U} = \sum_{e} \mathbf{K}_e\,{}^{t}\mathbf{U} = \sum_{e} {}^{t}\mathbf{F}_e \tag{3.26}$$

where e denotes the e^{th} element of the mesh and **F** is the elastic force. **F** for each "element" ($\mathbf{F}_e$)can be computed based on the element stiffness matrix and the nodal displacements of the element. After generating the vectors of the right hand side of Eq. 3.25, the large global Eq. 3.25 can be divided into simple independent equations, two for each node. Therefore, each node's displacement can be calculated separately using the following equation:

$${}^{t+\Delta t}U_i = \left(\frac{M_{ii}}{\Delta t^2} + \frac{D_{ii}}{2\Delta t}\right)^{-1} \left[{}^{t}R_i - F_i + \left(\frac{2M_{ii}}{\Delta t^2}\right) {}^{t}U_i - \left(\frac{M_{ii}}{\Delta t^2} - \frac{D_{ii}}{2\Delta t}\right) {}^{t-\Delta t}U_i\right] \tag{3.27}$$

for i=1, 2,. . . , 2N where i denotes the i^{th} component of the vector and ii denotes the diagonal entry in the i^{th} row of the matrix.

Chapter 4

Nonlinear Finite Element Modeling

The linear finite element formulation explained in the last chapter assumes infinitesimal displacements of the object and a linear elastic material. These assumptions, however, often do not apply to soft-tissues of human organs as discussed in Chapter 2. Nonlinear models are more suitable for simulating the deformation response for such soft-tissues. Nonlinear models can account for material and kinematic nonlinear effects that often characterize human organ soft-tissues deformations by employing suitable constitutive equations and large deformation stress and strain measures.

As it was mentioned in Chapter 2, total and updated Lagrangian formulations are the main formulations for nonlinear analysis [29]. In this thesis, the total Lagrangian formulation is used where stresses and strains are measured with respect to the original configuration. This choice allows for pre-computation of most spatial derivatives before the commencement of the time-stepping procedure and is capable of handling both geometric and material nonlinearities. Geometric nonlinearity is included by using a nonlinear strain measure in order to properly

model large deformations. The material nonlinearity is considered by using a nonlinear stress-strain relation based on the Ogden-based hyperelastic constitutive model [13, 42] as an energy function for the system. To achieve material incompressibility in the simulations, the energy function is modified by adding a term that would penalize volumetric (area in our case) variations in the deformed object.

The dynamic analysis is performed with an explicit time integration method as opposed to implicit methods. Compared to the formulation for implicit time integration, the development of the nonlinear formulation for explicit dynamic time integration is rather straightforward since it requires no iterations and no strain incremental stiffness matrices calculations. The only potential disadvantage of this approach is the conditional stability of the simulation which imposes a restriction on the size of the time step employed. The central difference method explained in Sec 3.2 is the explicit time integration employed.

The first section of this chapter provides an overview of the total Lagrangian formulation used in the nonlinear analysis to derive the discretized equations of motion. The second section introduces the stress and strain measures utilized in this thesis to enable modeling large deformations of the object. Finally, an illustration of the Ogden-based hyperelastic constitutive equation and the implementation of the incompressible condition is given in the last section. The material and notations used in this chapter are mainly based on [29].

4.1 Total Lagrangian Formulation with Explicit Time Integration

The FEM is a continuum-based approach in which the governing continuum mechanics equations are developed based on the principal of virtual work as discussed in Sec. 3.1.1. In this chapter, the same principal is employed with the addition of nonlinear effects including large displacement, rotations and strains as well as a nonlinear stress-strain relationship. Therefore, the principal of virtual work given in Eq. 3.5 and the analysis demonstrated in the previous chapter are the basis of the nonlinear analysis given below. Also, Fig. 3.1 can be employed in this chapter for the definition of the body under consideration since a similar notation is used.

In order to find the displacement field of a body corresponding to an applied load in nonlinear analysis, an incremental formulation is employed and time variables are used to describe the loading and motion of the body. The goal is to compute the displacement field of the body at discrete time points separated by an increment $\triangle t$. To obtain the solution at time $t + \triangle t$, the solution for all time steps from time 0 to time t are assumed to be known. This analysis is considered a Lagrangian formulation since all particles of the body are followed in their motion from the original to the current configuration.

The principal of virtual work for the body at time t in an incremental Lagrangian formulation should be considered since the dynamic analysis is performed with an explicit time integration method. Therefore, the principal of virtual work

equation (using tensor notation) is [29]:

$$\int_{{}^tA} {}^t\tau_{ij}\, \delta_t e_{ij}\, d^tA = {}^t\Re \tag{4.1}$$

where ${}^t\tau_{ij}$ is the cartesian components of the Cauchy stress tensor, $\delta_t e_{ij}$ is the strain tensor corresponding to virtual displacements, ${}^tA$ is the area at time t and ${}^t\Re$ is:

$${}^t\Re = \int_{{}^tA} {}^t f_i^b\, \delta u_i\, d\,{}^tA + \int_{{}^tB_f} {}^t f_i^B\, \delta u_i^B\, d\,{}^tl \tag{4.2}$$

where ${}^t f_i^b$ are components of external applied forces per unit area at time t, ${}^t f_i^B$ are components of externally applied boundary tractions per unit length at time t, ${}^tB_f$ is the boundary on which external tractions are applied, δu_i are components of virtual displacement vector and δu_i^B are the components of the virtual displacement vector evaluated on the boundary ${}^tB_f$. The components of δu_i corresponding to the prescribed displacement on the boundary ${}^tB_u$ are zero.

As in Chapter 3, the left hand side of the equation is the internal virtual work and the right hand side is the external virtual work ,but the configuration at time t is used. The strain tensor is given by:

$$\delta_t e_{ij} = \frac{1}{2}\left(\frac{\partial \delta u_i}{\partial\, {}^tx_j} + \frac{\partial \delta u_j}{\partial\, {}^tx_i}\right) \tag{4.3}$$

where ${}^tx_j$ are the cartesian coordinates of material points at time t and $j = 1, 2$. It can be seen that the components of the virtual displacement vector δu_i are functions of ${}^tx_j$ and the strain tensor is similar to the infinitesimal strain tensor in the linear analysis given in Eq. 3.2 except that the derivatives of the displacements are

with respect to the current coordinates. In the nonlinear analysis, the configuration is continuously changing which requires an incremental analysis and using appropriate strain and stress measures and constitutive relations that will be discussed in Sec. 4.2 and 4.3.

The large deformation stress and strain measures used in this thesis are defined in terms of the infinitesimal strain and the Cauchy stress measures later in Eqs. 4.21 and 4.19. Using these definitions, the following equality for the left hand side of Eq. 4.1 can be obtained [29]:

$$\int_{{}^tA} {}^t\tau_{ij}\, \delta_t e_{ij}\, d^tA = \int_{{}_rA} {}^t_rS_{ij}\, \delta\, {}^t_r\epsilon_{ij}\, d^rA \tag{4.4}$$

where ${}^t_rS_{ij}$ is the second Piola-Kirchhoff stress tensor and ${}^t_r\epsilon_{ij}$ is the Green-Lagrange strain tensor. The left superscript t indicates at which configuration the quantity occurs whereas the left subscript r indicates the reference configuration that the quantity is measured with respect to. The reference configuration can be any known configuration that was previously calculated from time 0 to time t. For the total Lagrangian (TL) formulation, the original configuration at time 0 is used as a reference. Therefore, for the TL formulation, Eq. 4.1 can be rewritten using Eq. 4.4 as follows:

$$\int_{{}_0A} {}^t_0S_{ij}\, \delta\, {}^t_0\epsilon_{ij}\, d^rA = {}^t\Re \tag{4.5}$$

Using Eq. 4.19, the right hand side of Eq. 4.5 can be written as follows [29]:

$$\int_{{}_0A} {}^t_0S_{ij}\, \delta_0 e_{ij}\, d^0A = {}^t\Re \tag{4.6}$$

The displacements of the material particles are the only variables in the equation of motion. In order to discretize the equation of motion given in Eq. 4.6 and obtain the governing FE equations, an analogy that is similar to the linear analysis given in Chapter 3 is employed. The body is divided into finite elements and the nodal displacements are solved for. Interpolation functions are used to interpolate the element displacements and coordinates.

Therefore, by using the element coordinate and displacement interpolations, Eq. 4.5 for a dynamic system can be written in a matrix form as follows:

$$\mathbf{M}\,{}^t\ddot{\mathbf{U}} + \mathbf{D}\,{}^t\dot{\mathbf{U}} + {}^t_0\mathbf{F} = {}^t\mathbf{R} \tag{4.7}$$

where $\mathbf{M}$ and $\mathbf{D}$ are the mass and damping matrices and are calculated based on the original configuration using Eqs. 3.18 and 3.19. ${}^t\mathbf{U}$, ${}^t\dot{\mathbf{U}}$, and ${}^t\ddot{\mathbf{U}}$ are the vectors of nodal displacements, velocities and accelerations, respectively. ${}^t_0\mathbf{F}$ is the nodal reaction forces vector and is given by:

$${}^t_0\mathbf{F} = \int_{{}^0A} {}^t_0\mathbf{B}_F^T\, {}^t_0\hat{\mathbf{S}}\, d^0A \tag{4.8}$$

where ${}^t_0\mathbf{B}_F$ is the full strain-displacement matrix , ${}^t_0\hat{\mathbf{S}}$ is the second Piola-Kirchhoff stress vector, and ${}^t\mathbf{R}$ is the external force vector. Based on the definition of external work given in Eq. 4.2, ${}^t\mathbf{R}$ can be written as:

$${}^t\mathbf{R} = \int_{{}^0A} \mathbf{H}^T\, {}^t_0\mathbf{f}^b\, d\,{}^0A + \int_{{}^0B_f} \mathbf{H}^{B^T}\, {}^t_0\mathbf{f}^B\, d\,{}^tl \tag{4.9}$$

where $\mathbf{H}$ is the interpolation function matrix, ${}^t_0\mathbf{f}^b$ is the body forces vector, and ${}^t_0\mathbf{f}^B$

is the boundary forces vector.

It should be noted that damping effects can be modeled in the constitutive equation when a strain-rate dependent material law is employed. However, since a hyperelastic material law that is not strain-rate dependant is considered in this thesis, damping effects are incorporated directly in the dynamic equation of motion in 4.7.

4.2 The Deformation Gradient and Large Deformation Stress and Strain Measures

In this section definitions of the strain and stress measures employed in the nonlinear analysis are given. These definitions start with the deformation gradient followed by the Green-Lagrange strain tensor as a long deformation strain measure. Finally the Second Piola-Kirchhoff stress tensor is introduced.

4.2.1 Deformation Gradient

To deal with a continuously changing configuration in a large deformation analysis, appropriate stress and strain measures must be developed . To this end, first the deformation gradient which relates the initial configuration at time 0 before the application of external loads to the deformed configuration at time t is defined as:

$$ {}^{t}_{0}\mathbf{X} = \begin{bmatrix} \frac{\partial^t x}{\partial^0 x} & \frac{\partial^t x}{\partial^0 y} \\ \frac{\partial^t y}{\partial^0 x} & \frac{\partial^t y}{\partial^0 y} \end{bmatrix} \tag{4.10} $$

where x and y are the components of the two-dimensional position vector $\mathbf{p}=[x\ y]^T$ of each particle in the body. Therefore if $d^0\mathbf{p}$ describes an infinitesimal fiber of material in the original configuration, the same fiber at time t $d^t\mathbf{p}$ is given by:

$$d^t\mathbf{p} = {}^t_0\mathbf{X}\, d^0\mathbf{p} \tag{4.11}$$

The deformation Gradient describes the stretches and rotations that the material undergo from time 0 to t and can be decomposed into two matrices as follows:

$${}^t_0\mathbf{X} = {}^t_0\mathbf{R}\, {}^t_0\mathbf{U} \tag{4.12}$$

where ${}^t_0\mathbf{R}$ is an orthogonal rotation matrix that represents rigid body rotations and ${}^t_0\mathbf{U}$ is a symmetric right stretch matrix that represents the stretches that the body has undergone. The eigenvalues of the right stretch matrix are the principal stretches and its eigenvectors are the principal stretch directions. These quantities will be used in the definition of the constitutive model and calculations of the stress vector that are explained in the next section.

4.2.2 Green-Lagrange Strain Tensor

The strain tensor is a measure of body deformation that is independent of rigid body motions. It indicates how much a length of a material has changed when going from the original configuration to the deformed configuration. The large deformation strain measure employed with the TL formulation is the Green-Lagrange strain tensor that is given by:

$${}^t_0\epsilon = \frac{1}{2}({}^t_0\mathbf{C} - I) \tag{4.13}$$

where ${}^t_0\mathbf{C}$ is the right Cauchy-Green deformation tensor which is defined in terms of the deformation gradient as follows:

$$ {}^t_0\mathbf{C} = {}^t_0\mathbf{X}^T \, {}^t_0\mathbf{X} \tag{4.14} $$

Using the polar decomposition of the deformation gradient given in Eq. 4.12, the right Cauchy-Green deformation tensor can be writhen in terms of the stretch tensor, i.e.

$$ {}^t_0\mathbf{C} = {}^t_0\mathbf{U} \, {}^t_0\mathbf{R}^T \, {}^t_0\mathbf{R} \, {}^t_0\mathbf{U} = {}^t_0\mathbf{U}^2 \tag{4.15} $$

which indicates that the eigenvalues of ${}^t_0\mathbf{C}$ are the square of the principal stretches and its eigenvectors are the principal directions.

The Green-Lagrange strain tensor can be compared with the small deformation tensor ${}^t_0\mathbf{e}$ by writing it in terms of displacements using the definition of deformation gradient in Eq. 4.10 and the fact that:

$$ {}^t p_i = {}^0 p_i + {}^t u_i \tag{4.16} $$

where ${}^t p_i$ and ${}^0 p_i$ are the current and original position vectors respectively, and ${}^t u_i$ is the corresponding deformation vector. The elements of the Green-Lagrange strain tensor are given by:

$$ \begin{aligned} {}^t_0\epsilon_{ij} &= \frac{1}{2}\left({}^t_0 X_{ki} \, {}^t_0 X_{kj} - \delta_{ij}\right) \\ &= \frac{1}{2}\left(\left(\delta_{ki} + \frac{\partial^t u_k}{\partial^0 p_i}\right)\left(\delta_{kj} + \frac{\partial^t u_k}{\partial^0 p_j}\right) - \delta_{ij}\right) \\ &= \frac{1}{2}\left(\frac{\partial^t u_i}{\partial^0 p_j} + \frac{\partial^t u_j}{\partial^0 p_i} + \frac{\partial^t u_k}{\partial^0 p_i}\frac{\partial^t u_k}{\partial^0 p_j}\right) \end{aligned} \tag{4.17} $$

where δ_{ij} is the kroneckor delta. It can be seen that all the derivatives are performed with respect to the original configuration and that Eq. 4.17 contains a quadratic term which makes the analysis nonlinear compared to the small deformation strain tensor which is a linear function of the deformation field,

$$ {}^{t}_{0}e_{ij} = \frac{1}{2}\left(\frac{\partial^t u_i}{\partial\, {}^0 p_j} + \frac{\partial^t u_j}{\partial\, {}^0 p_i}\right) \tag{4.18} $$

Based on the definitions of the Green-Lagrange strain tensor and small strain tensor in Eqs. 4.17 and 4.18, a variation in the current Green-Lagrange strain $\delta^t_0\epsilon$ is related to the variation in the small strain tensor $\delta_t\mathbf{e}$ by:

$$ \delta^t_0\epsilon_{ij} = \frac{\partial^t p_m}{\partial^0 p_i}\frac{\partial^t p_n}{\partial^0 p_j}\delta_t e_{mn} \tag{4.19} $$

This relation was used in Sec. 4.1 in order to derive the equality given in 4.4.

4.2.3 Second Piola-Kirchhoff Stress Tensor

The second Piola-Kirchhoff stress tensor ${}^t_0\mathbf{S}$ is the stress measure that is used in conjunction with the Green-Lagrange strain tensor. The relation between this long deformation stress tensor and the Cauchy stress tensor ${}^t\tau$ that is defined as the force per unit deformed area is given by the following equation [29]:

$$ {}^t\boldsymbol{\tau} = \frac{{}^t\rho}{{}^0\rho}\, {}^t_0\mathbf{X}\, {}^t_0\mathbf{S}\, {}^t_0\mathbf{X}^T \tag{4.20} $$

or in component form:

$$ {}^t\tau_{mn} = \frac{{}^t\rho}{{}^0\rho}\,\frac{\partial^t p_m}{\partial^0 p_i}\,\frac{\partial^t p_n}{\partial^0 p_j}\,{}^t_0S_{ij} \tag{4.21} $$

where ${}^t\rho$ is the mass density at time t and $\frac{{}^t\rho}{{}^0\rho} = det\ {}^t_0\mathbf{X} = {}^tJ$ which is the third invariant of the deformation gradient. Eq. 4.21 was used in Sec. 4.1 in order to derive the equality given in Eq. 4.4.

The second Piola-Kirchhoff tensor is a symmetric tensor that maps the force to the initial configuration of undeformed area. In case of hyperelastic models, which are employed in this thesis, the second PiolaKirchhoff stress can be evaluated as a derivative of an energy function with respect to the Green-Lagrange strain tensor [29] as will be shown in the next section.

4.3 Ogden-based Hyperelastic Constitutive Model

The constitutive equation enters the virtual work equation in the calculation of stresses to relate material stress to its strain. The TL formulation and the stress and strain measures covered in the last two sections include large deformation related nonlinear effects. However, in order to obtain a stress-strain relationship that can properly model the material, a specific constitutive relation should be employed. Appropriate constitutive models that account for the nonlinear properties of the soft-tissue need to be employed in order to have an accurate evaluation of the stresses based on strains in the body. This is essential for precise prediction of forces and deformation within the organ. The material in this section is based on [29,39,41,42].

The hyperelastic properties that are found in soft-tissues are usually modeled using the Ogden energy function [35,39]. This energy function is a generalization of the neo-Hookean and Mooney-Rivlin energy functions which are commonly

used in the literature. The Ogden material description is defined in terms of material constants and the principal stretches (a deformation measure) as follows:

$$ {}^{t}_{0}W = \sum_{n} \frac{\mu_n}{\alpha_n}(\lambda_1^{\alpha_n} + \lambda_2^{\alpha_n} - 2) \geq 0 \qquad where \quad \lambda_1\lambda_2 = 1 \tag{4.22} $$

where λ_i are the principal stretches which are the eigenvalues of the right stretch tensor as explained in Sec. 4.2 and will be illustrated in Eq. 6.24; μ_n and α_n are material constants.

The Ogden energy function reduces to the neo-Hookean (for n=1 and α_1=2)and Mooney-Rivlin (for n=2, α_1=2 and α_1=-2) energy functions. The extra flexibility in choosing α and n in the Ogden model can potentially provide more accurate results. However, choosing n to be higher than 1 can cause complexity in the calculations and difficulty in fitting the material constants from experiments. It can be seen from Eq. 4.22 that the energy in the body is equal to zero when there is no deformation, i.e. when the principal stretches are equal to 1. The principal stretches are higher than 1 in the case of tension and below 1 in the case of compression. Also, the function is always positive which is a requirement for an energy function.

The Ogden function given in Eq. 4.22 is for two-dimensional planar analysis. The constrain on the principal stretches is to enforce a totally incompressible property since the term $\lambda_1\lambda_2$ is the determinant of the deformation gradient that defines the ratio of the deformed area over the initial area. Therefore, a ratio equal to 1 indicates that no change in area has occurred while deforming. Soft-tissues

are considered a type of a rubber-like material that behave as almost incompressible [29,39]. Therefore, a better assumption is only an almost incompressible material that is modeled by modifying the stain energy function given in Eq. 4.22 to include the constrain as a penalty term with the bulk modulus κ as a Lagrange multiplier (hydrostatic work term) as follows:

$$ {}^t_0\bar{W} = \sum_n \frac{\mu_n}{\alpha_n} \left(\frac{\lambda_1^{\alpha_n} + \lambda_2^{\alpha_n}}{(\lambda_1\lambda_2)^{\frac{\alpha_n}{2}}} - 2 \right) + \frac{1}{2}\kappa({}^t_0J_3 - 1)^2 \tag{4.23} $$

where ${}^t_0J_3$ is the third reduced invariant of the right Cauchy-Green deformation tensor ${}^t_0\mathbf{C}$ and is defined as:

$$ {}^t_0J_3 = ({}^t_0I_3)^{\frac{1}{2}} = (det{}^t_0\mathbf{C})^{\frac{1}{2}} = (\lambda_1^2\lambda_2^2)^{\frac{1}{2}} = \lambda_1\lambda_2 \tag{4.24} $$

It should also be pointed out that the principal stretches λ_i are replaced with $\lambda_i(\lambda_1\lambda_2)^{-1/2}$ in order to make the terms under the summation unaffected by volumetric deformations. The bulk modulus κ should be several thousands times the shear modulus in order to model an almost incompressible behavior [29].

In this thesis, the Ogden strain energy function with n=1 is employed as it was done in [13]. However, the almost incompressibility property is enforced in this thesis (as opposed to [13]) by modifying the Qgden energy function through adding a hydrostatic work term that penalizes the change in area. Therefore the employed Ogden-based energy function is:

$$ {}^t_0\bar{W} = \frac{\mu_1}{\alpha_1} \left(\frac{\lambda_1^{\alpha_1} + \lambda_2^{\alpha_1}}{(\lambda_1\lambda_2)^{\frac{\alpha_1}{2}}} - 2 \right) + \frac{1}{2}\kappa({}^t_0J_3 - 1)^2 \tag{4.25} $$

Using the definition of the initial shear modulus in terms of these material constants,

$$\mu = \frac{1}{2}\sum_{i=1}^{n} \alpha_i \mu_i \tag{4.26}$$

μ_1 can be expressed in terms of the initial shear modulus as follows:

$$\mu_1 = \frac{2\mu}{\alpha_1} \tag{4.27}$$

By substituting Eq. 4.27 into Eq. 4.25, the following Ogden energy function is obtained:

$${}^{t}_{0}\bar{W} = \frac{2\mu}{\alpha_1^2}\left(\frac{\lambda_1^{\alpha_1} + \lambda_2^{\alpha_1}}{(\lambda_1\lambda_2)^{\frac{\alpha_1}{2}}} - 2\right) + \frac{1}{2}\kappa({}^{t}_{0}J_3 - 1)^2 \tag{4.28}$$

This energy function is used to calculate the second Piola-Kirchhoff stresses by evaluating its derivative with respect to the Green-Lagrange strain tensor:

$${}^{t}_{0}S_{ij} = \frac{\partial \bar{W}}{\partial {}^{t}_{0}\epsilon_{ij}} \tag{4.29}$$

It is also possible to find the derivatives of the energy function with respect to the principal stretches λ_i using the chain rule which yields the principal second Piola-Kirchhoff stresses, i.e.:

$${}^{t}_{0}S_i = \frac{\partial \bar{W}}{\partial {}^{t}_{0}\epsilon_i} = \frac{\partial \bar{W}}{\partial {}^{t}_{0}\lambda_i}\frac{\partial {}^{t}_{0}\lambda_i}{\partial {}^{t}_{0}\epsilon_i} = \frac{1}{{}^{t}_{0}\lambda_i}\frac{\partial \bar{W}}{\partial {}^{t}_{0}\lambda_i} \tag{4.30}$$

since ${}^{t}_{0}\epsilon_i = \frac{1}{2}({}^{t}_{0}\lambda_i^2 - 1)$. It should be noted that the right subscript i in Eq. 4.30 denotes the i_{th} principal stress value.

By using the Ogden-based energy function given in Eq. 4.28 in Eq. 4.30, the

principal second Piola-Kirchhoff stresses are:

$$ {}^{t}_{0}S_1 = \frac{1}{{}^{t}_{0}\lambda_1}\frac{\partial \bar{W}}{\partial {}^{t}_{0}\lambda_1} = \frac{2\mu}{\lambda_1 \alpha^2}\left(\frac{\alpha}{2}\lambda_1^{\frac{-\alpha}{2}}\lambda_2^{\frac{-\alpha}{2}} - \frac{\alpha}{2}\lambda_1^{\frac{-3\alpha}{2}}\lambda_2^{\frac{\alpha}{2}}\right) + \kappa(\lambda_1\lambda_2 - 1)\frac{\lambda_2}{\lambda_1} \tag{4.31} $$

$$ {}^{t}_{0}S_2 = \frac{1}{{}^{t}_{0}\lambda_2}\frac{\partial \bar{W}}{\partial {}^{t}_{0}\lambda_2} = \frac{2\mu}{\lambda_2 \alpha^2}\left(\frac{-\alpha}{2}\lambda_1^{\frac{\alpha}{2}}\lambda_2^{\frac{-3\alpha}{2}} + \frac{\alpha}{2}\lambda_1^{\frac{-\alpha}{2}}\lambda_2^{\frac{-\alpha}{2}}\right) + \kappa(\lambda_1\lambda_2 - 1)\frac{\lambda_1}{\lambda_2} \tag{4.32} $$

In summary, the known values of the principal stretches for the configuration at time t are used to calculate the principal second Piola-Kirchhoff stress values which in turn are employed to compute the nodal reaction force vector by Eq. 4.8. Also, as it was mentioned in the beginning of the chapter, the central difference method is utilized to solve the governing equations given in Eq. 4.7 and find the displacement field at time $t + \triangle t$ using Eq. 3.27 as explained in Sec 3.2.

Chapter 5

Modeling of Soft-tissue Cutting

Soft-tissue cutting is one of the most complicated surgical tasks to model. It changes the way the elements are joined in the mesh. This is known as a topological modification. Many techniques have been used to model topological modifications as discussed in Chapter 2. An element separation technique is developed in this thesis that does not increase the number of elements in the model, preserves the mass and area of the model, and creates a cut that is closest to the path traversed by the tool using node snapping. The cutting technique that is presented here uses the Ogden-based hyperelastic nonlinear mathematical model with the almost incompressible condition that was presented in the last chapter in order to take into account material and geometrical nonlinearities of soft-tissues.

A progressive cutting algorithm is developed by allowing the mesh to deform until a cutting criterion, that will be introduced later in this chapter, is met. Therefore, at each time step, the cutting criterion is checked; when the condition is met, a cut is introduced to the FE mesh through the following steps that are also shown in Fig. 5.1:

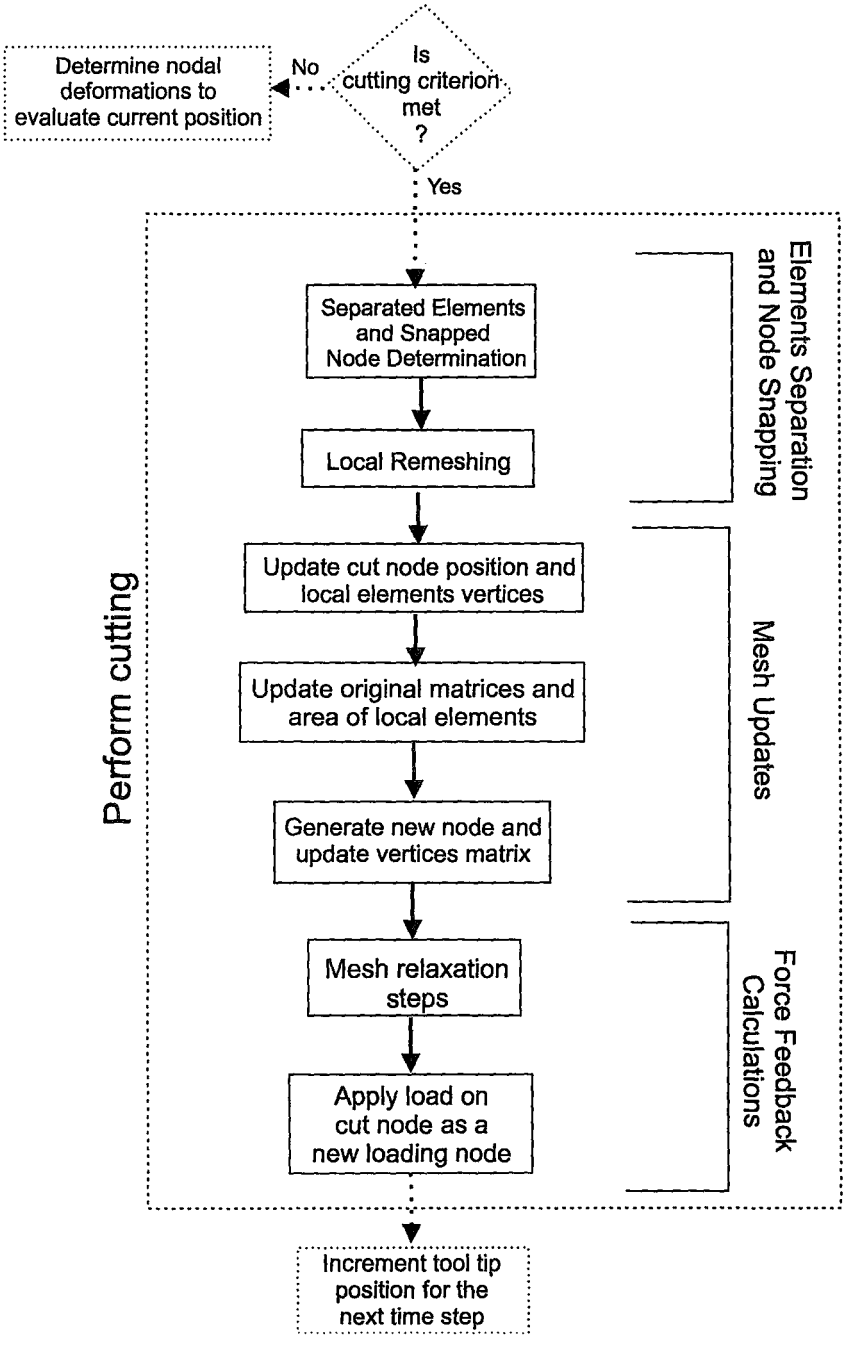


Figure 5.1: Flow chart of the algorithm employed to perform cutting

- **Elements separation and node snapping**: Upon meeting the cutting criterion, the next node that is closest to the cut path is determined and snapped to the path. This is achieved by projecting the node into the cut path in order to have a cut that best fits the tool direction. In some cases, node snapping could generate badly shaped elements or elements with very small area that could cause numerical instability in the simulation. To avoid that, the local cutting area is remeshed by retriangulating the elements and allowing the snapped node to move if necessary in order to keep the element's edge lengths bigger than a threshold value that is specified by the time step. This will eliminate the necessity of decreasing the time step when the element's area becomes smaller than a threshold value.

- **Mesh updates**: After performing the local remeshing, the original matrices and areas of the remeshed elements are updated using an inverse parametric method that is based on shape functions in order to find the original position of the snapped node. The cut is introduced by splitting the load node into two nodes with each one having half the mass of the original node and the same original and current position. Also, the nodal reaction forces between the separated elements disappear by updating the triangle's vertices in order to make the elements on one side of the cut path have the original node number and the elements on the other side have the new node number.

- **Force feedback calculations**: To generate a haptic feedback that is similar to what is obtained in experiments shown in the literature [1,66], few time steps are needed for mesh relaxation after node snapping. The mesh relaxation is performed locally by letting the mesh deform without an external force on

the load node. This step allows the forces to decrease after a cut is performed and then the forces start to increase again in the following time steps when the load is applied on the snapped node. In order to avoid overlapping of the nodes on the cut path, a constrain on these nodes' position is added by forcing the nodes to stay on one side of the cut path. The tool is assumed to have a known trajectory.

The following sections include a detailed explanation of each step of the algorithm briefly described above.

5.1 Cutting Criterion

To determine when a cut should be created in the soft-tissue model, the external force exerted on the the tissue model is used as a cutting criterion. A similar approach has been employed in [66,67]. Determining the threshold value for cutting is straight forward with this approach since soft-tissue cutting experiments presented in the literature [1,66] measure the force on the tip of the tool. If the external force exceeds the maximum load that the soft-tissue can resist then a cut should be executed. This maximum external force is dependent on the material property that would indicate its toughness. Also, it is possible to make the critical external force value dependent on the cutting tool sharpness, tool velocity and previous damage in the cutting area as it was explained in Sec. 2.3.

After calculating the external force on the contact node between the tool tip and the FE mesh (load node) in each time step, this force is decomposed into two components relative to the tool which is modeled as a line since a two-dimensional

analysis is performed. One component is along the tool direction, whereas the second component is perpendicular to this direction. The first is used as a physical criterion for cutting. Therefore, when the external force in the tool direction reaches a threshold value, the modeled object should be broken apart and the cutting procedure is performed by carrying out the following steps in order to separate elements and change the topology of the FE mesh.

5.2 Elements Separation and Node Snapping

5.2.1 Determination of Separated Elements and Snapped Node

Since element separation technique is used to create the cut in the FE mesh, the two triangular elements (cut elements) that must be separated should be determined first. These cut elements are the two elements that share the cut edge between the current contact node (load node) where the load is being applied and the next contact point at which the load will be applied after the cut is performed. The second node of the cut edge (cut node) is chosen based on the angle that the cut edge makes with the cutting tool trajectory. Therefore, the edge that makes the smallest angle with the cutting tool trajectory is chosen as the cut edge and the two elements that share this edge are the cut elements that will be separated. This process is illustrated in Fig. 5.2 where $\theta_1 < \theta_2$. Boundary edges are excluded from the possibility of being a cut edge in order to avoid changing the overall area of the object when node snapping is performed.

In order to have a smooth cut is as close as possible to the cutting trajectory, the

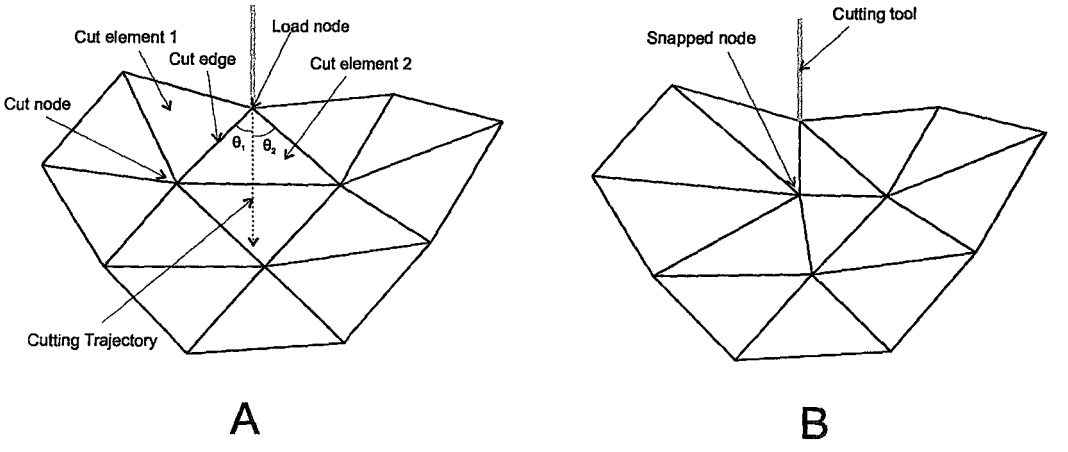


Figure 5.2: Illustration of determination of separated elements and node snapping: (A) Before node snapping. (B) After node snapping

second node on the cut edge (cut node) should be snapped to the line that represents the tool trajectory. This is achieved by determining the orthogonal projection of the cut node on the cutting trajectory and moving the cut node to that point (see Fig. 5.2). It should be noted that the cut node is the node where the load will be applied after the cut. Therefore the tool tip position should be offset to the orthogonal projection point where the cut node is placed in order to apply the load on this node in the following time steps.

5.2.2 Local Remeshing

Node snapping that is performed in the previous step could generate badly shaped elements or elements with very small area which are called degenerated triangles as shown later in Figs. 5.5 , 5.6, and 5.7. These elements can cause numerical instability in the simulation because of their small characteristic length which would make the critical time step for guaranteed lower than the employed time

step [14, 29]. Therefore, to avoid the necessity of decreasing the employed time step due to degenerated elements, the local cutting area which includes the elements that surround the snapped node is remeshed. The remeshing algorithm retriangulates the elements and allows the snapped node to move if needed without creating any new elements in the mesh. The local remeshing algorithm employed in this thesis is a modification of the meshing algorithm given in [71] and is shown in Fig. 5.3. The area that is remeshed is a polygon specified by the nodes of the elements in contact with the snapped cut node. The boundary nodes of these elements are fixed. The steps of the algorithm includes the following:

1. The area specified by the nodes is triangulated using the Delaunay function which returns a set of triangles such that no data points are contained in any triangle's circumscribed circle [71]. The Delaunay triangulation maximizes the minimum angle of all the angles of the triangles in the triangulation.

2. The quality of the triangular elements created are calculated as a measure of the size and shape of the elements. This is employed to detect the degenerated elements (e.g. see Fig 5.4) that can cause numerical instability because of small edge length or small area. As it was mentioned before, the characteristic length of the element determines the critical time step. Also, as the area of the element tends to zero, force computations give infinite deformations that indicate numerical instability [49]. Many quality measures are available in the literature [72]. The quality measure employed in this thesis is:

$$quality(element) = 4\sqrt{3}\frac{\triangle}{(L_1^2 + L_2^2 + L_3^2)} \tag{5.1}$$

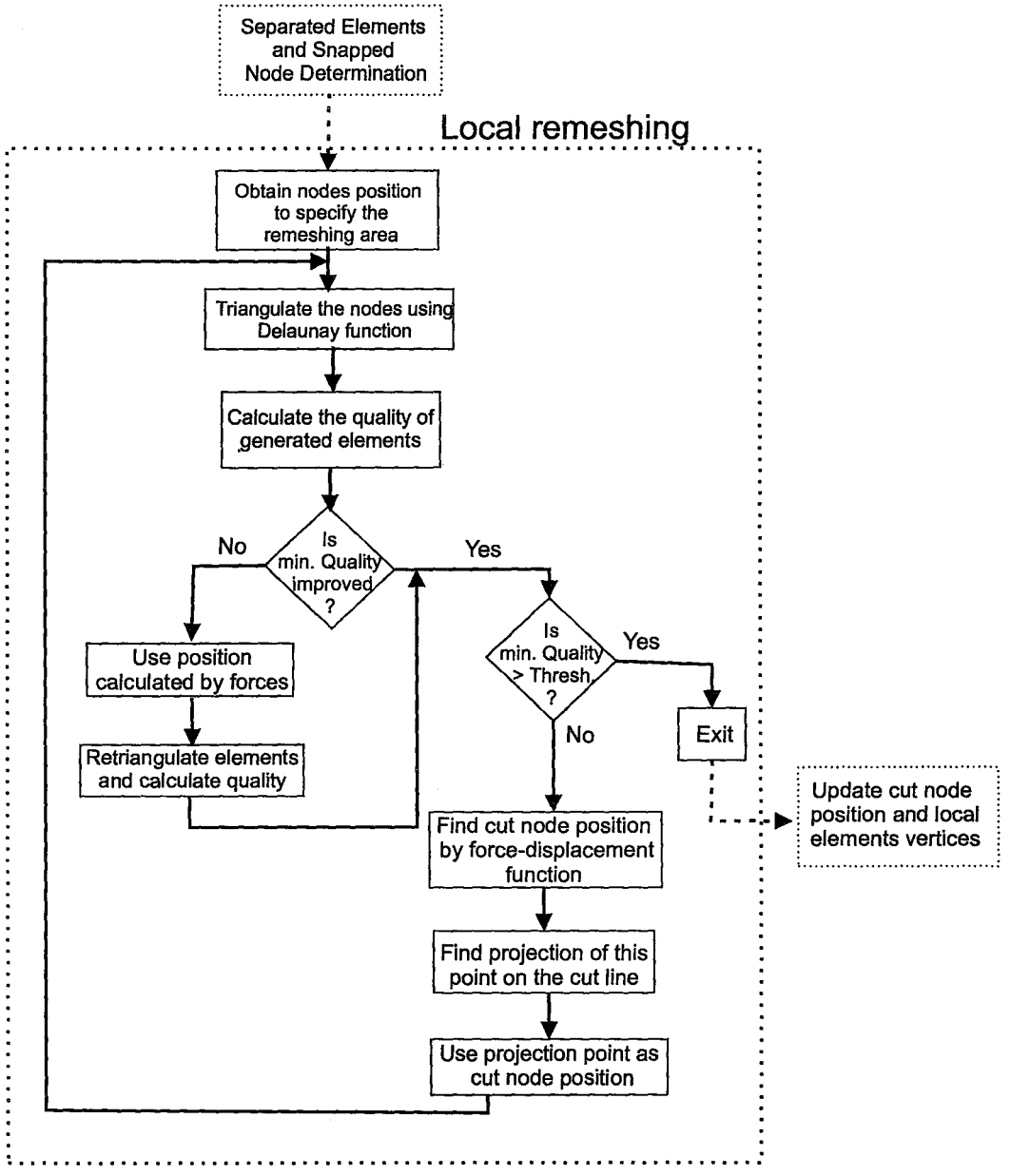


Figure 5.3: Flow chart of the local remeshing algorithm

Figure 5.4: Examples of degenerated triangular elements: (A) Small edge length. (B) Small area.

where $\triangle$ is the area of the triangular element and L_i is the length of each edge in the triangle. The constant is employed to normalize the measure for equilateral triangles and makes the measure take on values between zero for complete degenerated elements and one equilateral elements. Therefore, degenerated elements that can cause numerical instability have low quality measure.

3. The calculated quality of each element is compared with a threshold value that is known to satisfy the stability condition of the explicit integration method. If the quality of all elements are above the threshold value then the algorithm exits with the triangles obtained from the Delaunay function which can be a retriangulation of the elements in the original mesh as shown in Fig. 5.5. Otherwise, the internal node, which is the snapped cut node, is repositioned using a linear force-displacement relationship to solve for an equilibrium in a truss structure [71] as follows:

$$\mathbf{p}_{n+1} = \mathbf{p}_n + \triangle t_E \, \mathbf{F}(\mathbf{p}_n) \tag{5.2}$$

where $\mathbf{p}_{n+1}$ and $\mathbf{p}_n$ is the current and previous position of the snapped node and $\triangle t_E$ is the time step in Eulers method. $\mathbf{F}(\mathbf{p}_n)$ is the force excreted on the snapped node based on the previous position and it is the sum of the

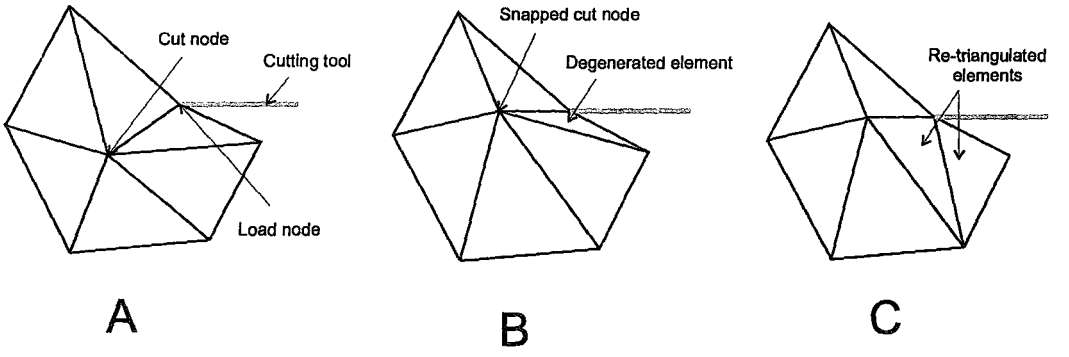


Figure 5.5: Example to illustrate local remeshing with retriangulation: (A) Before node snapping. (B) Degenerated elements creation after node snapping. (C) Retriangulation of local elements.

repulsive force vectors from all edges connected to the node:

$$\mathbf{F}(\mathbf{p}_n) = \sum_{edge} f_{edge} \tag{5.3}$$

The repulsive force of each edge f_{edge} depends on the edge current length ℓ and the unextended length ℓ_0 as follows:

$$f_{edge}(\ell, \ell_0) = \begin{cases} k(\ell_0 - \ell) & \text{if } \ell < \ell_0 \\ 0 & \text{if } \ell \geq \ell_0 \end{cases} \tag{5.4}$$

Therefore, the resultant force $\mathbf{F}(\mathbf{p}_n)$ moves the snapped node in order to equalize the length of the edges connected to the snapped node after few iterations. This process will elongate the short edges which improves the quality of the degenerated elements as shown in Fig. 5.6.

4. In some cases, the node repositioning using the force generated based on the length of the edges is away from the cutting tool trajectory and sometimes

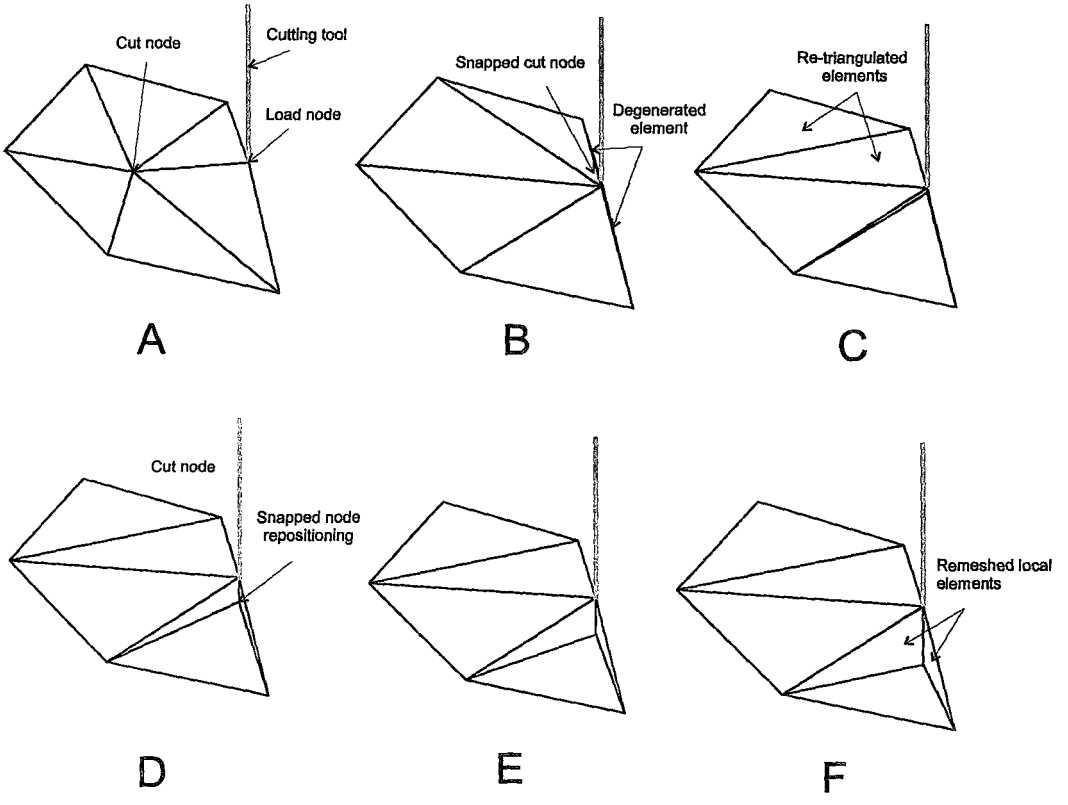


Figure 5.6: Example to illustrate local remeshing with snapped node repositioning: (A) Before node snapping. (B) Degenerated elements creation after node snapping. (C,D,E,F) Local elements remeshing by retriangulation and snapped node repositioning.

moving the node away from the cut line is not necessary to improve the quality of the element as it is shown in Fig. 5.7C. Therefore, in order to have a cut that is as close as possible to the cutting profile whenever it is possible, the orthogonal projection of the new position on the cutting tool trajectory is found and it is assumed to be the new position of the cut node as shown in Fig. 5.7D. If the projection point improves the quality of the elements, it is taken as the new position of the snapped node; whereas, if it worsens the quality of the elements, the position calculated using the repulsive forces as explained in the previous step is taken as the new position of the snapped node. With this approach, the snapped node will be placed on the cut trajectory whenever it is possible in order to create a cut that is as close as possible to the intended cutting path.

5.3 Mesh Updates

After remeshing the local cutting area, the following updates should be performed:

1. The position of the snapped cut node is updated with the value obtained from the local remeshing algorithm, which will guarantee the stability of the simulation in the subsequent time steps. The vertices of the local remeshed elements should be updated because the Delaunay function can change the topology of these elements by retriangulation as it was shown in Figs. 5.5 and 5.6.

2. In order to update the original matrices and areas of the remeshed elements, the new original position of the snapped node ${}^0\mathbf{p}_{new}$ should be found since

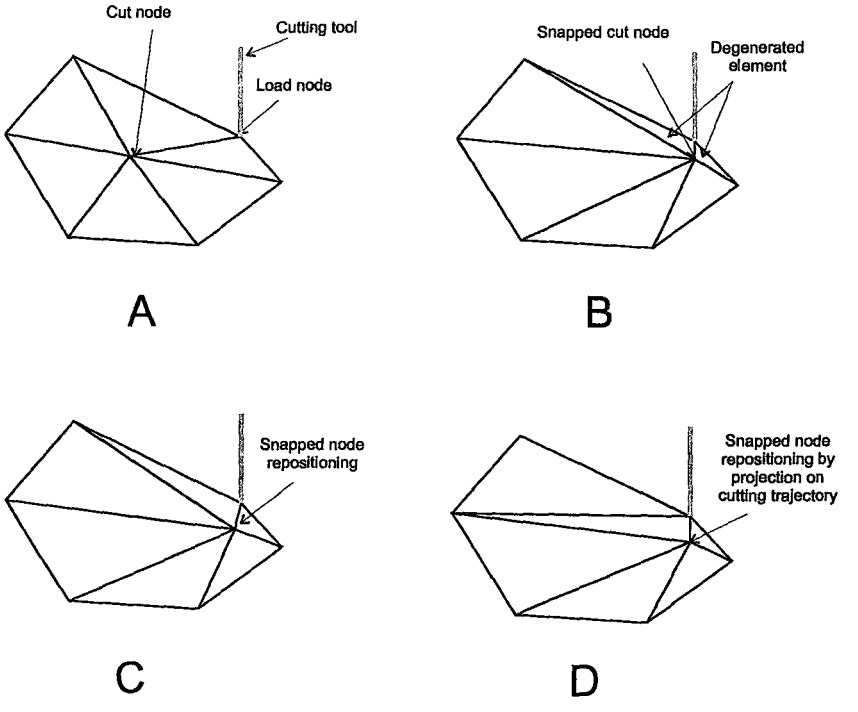


Figure 5.7: Illustration of local remeshing by snapped node repositioning with orthogonal projection: (A) Before node snapping. (B) Degenerated elements creation after node snapping. (C) Local elements remeshing by snapped node repositioning (D) Local elements remeshing by projecting the new snapped node on the cutting trajectory.

its new position ${}^t\mathbf{p}_{new}$ is calculated based on the current deformed configuration as it was explained above in Sec. 5.2. The inverse isoparametric mapping method mentioned in [63,73,74] is employed to find the original position of the snapped node. This method involves first finding the element that contains the new point by utilizing the polygon bounding box method and then obtaining the isoparametric coordinates (h_1, h_2, h_3) of the new point with respect to the three nodes of this element. As it was explained in 3.1.2, these coordinates define the interpolation function of the new point and they are

found using the current position of the three nodes of the element that contains the new point using the following three equations:

$$ {}^{t}x_{new} = h_1 \, {}^{t}x_1 + h_2 \, {}^{t}x_2 + h_3 \, {}^{t}x_3 \tag{5.5} $$

$$ {}^{t}y_{new} = h_1 \, {}^{t}y_1 + h_2 \, {}^{t}y_2 + h_3 \, {}^{t}y_3 \tag{5.6} $$

$$ h_1 + h_2 + h_3 = 1 \tag{5.7} $$

where Eqs. 5.8 and 5.6 are based on Eq. 3.9 and show how the x and y coordinates of a point in an element is defined based on the x_i and y_i coordinates of the three nodes in the element using the isoparametric coordinates. Eq. 5.7 is a property of the isoparametric coordinates. All nodes coordinates ($^{t}x_i$ and $^{t}y_i$) and the new point coordinates ($^{t}x_{new}$ and $^{t}y_{new}$) are known in the deformed configuration; therefore, the isoparametric coordinates can be found by solving Eqs.(5.8)-(5.7). Therefore, the new original position of the snapped node $^{0}\mathbf{p}_{new}$ can be computed using the obtained isoparametric coordinates and the original positions of the element's three nodes $^{0}\mathbf{p}_i$ as follows:

$$ {}^{0}\mathbf{p}_{new} = \sum_{i=1}^{3} h_i \, {}^{0}\mathbf{p}_i \tag{5.8} $$

where $\mathbf{p} = [x \; y]^T$. The displacement field ,$^{t}\mathbf{U}$ and $^{t-\Delta t}\mathbf{U}$ of the snapped node should also be mapped from the old to the new position by employing the inverse isoparametric mapping method as well based on Eq. 3.8.

The area $\triangle$ and elemental strain-displacement matrix $\mathbf{B}_m$ mentioned in Eq. 6.5

should be recalculated for the remeshed local elements using the new original position of the snapped cut node. The details of how these calculations are performed will be explained in 6.1.1. It should be noted that the mass of the nodes in these elements are not updated according to the area of the elements. This is to avoid having small masses at nodes connected to small elements that could potential cause numerical instability. instead, the same mass matrix that is generated based on the original configuration is employed.

3. To separate the two elements and create a cut in FE mesh, a new node is established by splitting the load node into two nodes with each one having half the mass of the original node and the same original and current position. Also, in order to eliminate the nodal reaction forces between the separated elements, triangles' vertices matrix is updated to make the elements on one side of the cut trajectory have the original node number and the elements on the other side have the new node number.

5.4 Force Feedback Calculations

Typical soft-tissue cutting experiments found in the literature [1,66] show that the force at tip of the surgical tool drops down when a cut is created in the tissue and when a further loading is applied, the force starts increasing again as shown in Fig. 5.8. In order to have a similar behavior in the modeling process and generate a haptic feedback that is comparable with experimental results reported in the literature, the following steps are performed:

1. A number of time steps are needed for mesh relaxation after node snapping and local remeshing are performed. The mesh relaxation is done locally on the remeshed elements only. The nodes of these elements are set to deform freely to the relaxed position without any external force applied on the load node. Therefore the position of these nodes are updated after each time step and a higher damping is used during these steps in order to reach the relaxed position without many oscillations. It should be noted that these time steps are internal and are not used for the actual deformation and force calculations. They only allow for decreasing the high forces generated after the remeshing when a cut is performed because of the sudden big change in the snapped node position in one time step.

2. Preparing for the next time step, a load is applied on the cut node as a new load node. The response of the whole object is obtained by calculating the deformation field given the applied external force.

After calculating the new position of nodes in each time step,the position of the nodes on the cut line is examined. To avoid the overlapping of these node after a cut, a constrain on the position of these nodes is added to force them to stay on the same side of the cut path all the time. Thus, if it is found that a node has crossed the cut path, then its position is changed to the orthogonal projection of the node position on the cut path.

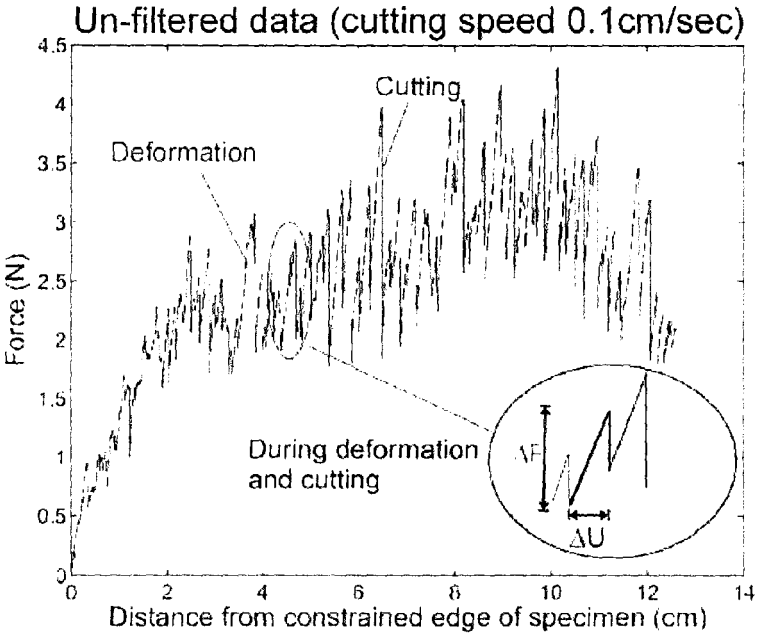


Figure 5.8: Experimental data from liver cutting. Courtesy of [1]

Chapter 6

Deformation and Cutting Simulation: Algorithms and Results

This chapter discusses the implementation of the deformation/cutting modeling algorithms and presents the results of numerical simulations to evaluate the performance of the proposed modeling techniques. Based on the material of Chapter 3, the first section shows the detailed computations that need to be carried out when a linear FE analysis is employed with the explicit time integration method. An overview of the computations involved in the nonlinear FEM algorithm is given in the second section, which is essentially an extension of Chapter 4. The third section gives a summary of the overall algorithm employed to simulate successive soft-tissue cutting based on the steps given in Chapter 5. Each of the sections also contains the results of numerical simulations conducted with the models employed.

6.1 Steps and Results of 2D Explicit Dynamic Linear Finite Element Algorithm

The linear analysis is included in this thesis for comparison the nonlinear FE model. The steps of the algorithm for linear FEM analysis can be divided into two main groups. The first group includes calculations that can be performed off-line prior to the start of the simulation whereas the second group includes on-line computations that must be conducted at each time step. Fig. 6.1 shows the main blocks of the algorithm and below are the details of what is performed at each step.

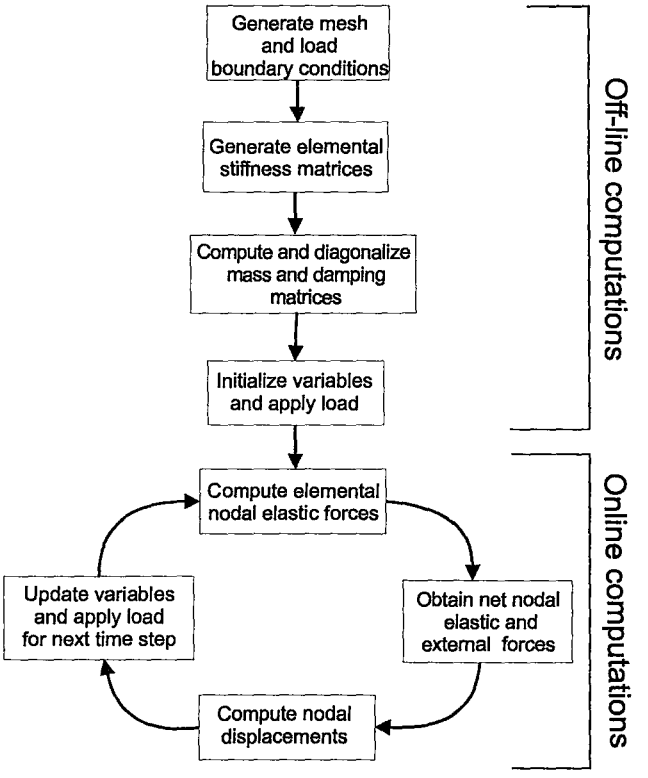


Figure 6.1: Steps of 2D linear dynamic finite element algorithm

6.1.1 Off-line Precomputations and Initializations

The following calculations can be performed prior to the start of the simulation:

Generate the Mesh and Specify Load and Boundary Condition Nodes

The available MATLAB source code of the mesh generator presented in [71] is used to generate the mesh of a 2D object. To this end, the geometry of the object should be defined as a distance function that returns the signed distance from each node position to the closest boundary. The distribution of elements in the mesh whether uniform or non-uniform can be specified by an edge length scaling function. The required resolution of the mesh (fine or coarse) is determined by a number that represents the initial edge length in the mesh. In this thesis, without loss of generality, the 2D object was defined as a circle with a radius of $0.1m$ which is representing a slice of a 3D deformable object. The object is uniformly meshed with triangle elements that have an approximate edge length of $0.02m$ as it is shown in Fig. 6.2. All interaction forces and deformations occur in the 2D plane.

The output of the meshing function is two matrices. The first matrix (P) specifies the nodes positions. The size of this matrix is (Nx2) where N is the number of nodes. In each row of the matrix, the first element is the x coordinate of the corresponding node position (row number) and the second element is the y coordinate of that node position. The second matrix (T) consists of the indices (nodes) of each element. Since the triangular elements are utilized, each element has 3 nodes. Therefore, the size of this matrix is (NEx3) where NE is the number of triangle elements.

To prevent the object from moving freely, a sufficient number of nodes are

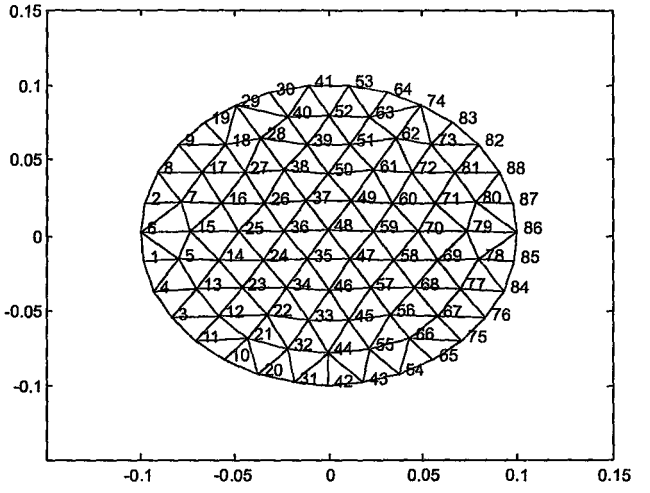


Figure 6.2: Meshed circle with the designated node numbers

fixed. These nodes are called boundary condition nodes and act as a support to the meshed object. The number of the nodes that are needed to be fixed should be specified by the user. At the same time, the node number where the load is applied should also be provided.

Generate the Elemental Stiffness Matrices

Since a linear elastic system is being solved, the stiffness matrix is constant through out the simulation. Therefore, the elemental stiffness matrices can be calculated in the precomputation stage.

As it was shown before in Eq. 3.14, the stiffness matrix for each element can be calculated using the following integral:

$$\mathbf{K}_m = \int_{A_m} \mathbf{B}_m^T \mathbf{C}_m \mathbf{B}_m \mathrm{d}A_m \tag{6.1}$$

where $\mathbf{B}_m$ is the element strain-displacement matrix, $\mathbf{C}_m$ is the element matrix of material constants. Since these matrices are constant for a specific element, the integral is performed on the area of each element and the elemental stiffness matrices are computed according to the following equation:

$$\mathbf{K}_m = \mathbf{B}_m^T \mathbf{C}_m \mathbf{B}_m \triangle \tag{6.2}$$

where $\triangle$ is the area of each triangular element. Therefore, in order to find the stiffness of each element, $\mathbf{B}_m$, $\mathbf{C}_m$ and $\triangle$ have to be calculated for each element as follows:

- $\mathbf{C}_m$ calculation: Since the object is considered to have one type of material, $\mathbf{C}_m$ is the same for all elements in the mesh and needs to be calculated only once. The material matrix for an isotropic material when plane strain analysis for a 2D slice of a unit thickness is being performed is given by [29,56]:

$$\mathbf{C}_m = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix} \tag{6.3}$$

  where ν is the Poisson's ratio and E is the Young's modulus which are the parameters that specify the properties of the material. A Poisson's ratio of ν=0.49999 that represents an almost incompressible material and a Young's modulus of (E=1800 Pa) are utilized in the simulation.

- $\triangle$ calculation: The area of each triangle is calculated as follows:

$$2\triangle = det \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \tag{6.4}$$

where x and y are the coordinates of nodes 1, 2, and 3 of each element. Therefore, the P and T matrices that were generated by the meshing algorithm can be used to calculate the area for each element as well as the B matrix of each element as explained in the next step.

- $\mathbf{B}_m$ matrix calculation: This matrix needs to be calculated for each element separately because it depends on the cartesian coordinations of the nodes of each element as it will be shown below. Based on the strain definition in Eqs. 3.2 and 3.10, and Eq. 3.6 that shows the calculation of the displacement field using the element nodal displacements, the $\mathbf{B}_m$ matrix can be computed as follows:

$$\mathbf{B}_m = \mathbf{L}\mathbf{H}_m = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \begin{bmatrix} h_1 & 0 & h_2 & 0 & h_3 & 0 \\ 0 & h_1 & 0 & h_2 & 0 & h_3 \end{bmatrix} \tag{6.5}$$

where $\mathbf{L}$ is a linear operator matrix, $\mathbf{H}_m$ is the element shape function matrix and h_k is the shape function of each node of the element. For the linear triangle element, this function is given by:

$$h_k = \frac{a_k + b_k x + c_k y}{2\triangle} \tag{6.6}$$

where k=1, 2, and 3 correspond to the local node number for each element. Therefore, by substituting Eq. 6.6 in Eq. 6.5, the B matrix for each element is given by:

$$\mathbf{B}_m = \frac{1}{2\triangle}\begin{bmatrix} b_1 & 0 & b_2 & 0 & b_3 & 0 \\ 0 & c_1 & 0 & c_2 & 0 & c_3 \\ c_1 & b_1 & c_2 & b_2 & c_3 & b_3 \end{bmatrix} \tag{6.7}$$

where

$$b_1 = y_2 - y_3 \qquad c_1 = x_3 - x_2$$

$$b_2 = y_3 - y_1 \qquad c_2 = x_1 - x_3$$

$$b_3 = y_2 - y_2 \qquad c_3 = x_2 - x_1$$

Based on Eq. 6.2, the size of the stiffness matrix generated for each element is a 6×6 given that the elements have 3 nodes and each node has 2 coordinates.

Compute, Diagonalize and Assemble Elemental Mass and Damping Matrices

The mass matrix of each element is calculated using the following integral which was derived in 3.18:

$$\mathbf{M}_m = \int_{A_m} \rho_m \mathbf{H}_m^T \mathbf{H}_m \mathrm{d}A_m \tag{6.8}$$

Using the linear shape functions of the triangle given in Eq. 6.6, the left hand side of Eq. 6.9 is obtained. In order to have a diagonal mass matrix which is needed to simplify the computations when explicit time integration method is employed, the row sum method is used [56]. In this method, the elements of each row are added and the sum is placed in the diagonal entry of that row. The lumped mass

matrix is shown in the right hand side of the following equation:

$$\mathbf{M}_m = \frac{M}{12}\begin{bmatrix} 2 & 0 & 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 \\ 1 & 0 & 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 1 & 0 & 2 \end{bmatrix} \xrightarrow[\textit{Row sum method}]{\textit{Diagonalize}} \mathbf{M}_m = \frac{M}{3}\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.9}$$

where M=total mass of the element = $\rho_m \triangle$, ρ_m= mass density of the element and all elements are assumed to have the same mass density, and $\triangle$=area of each triangular element.

To generate the system mass matrix, the elemental mass matrices can be assembled by adding the contribution of each element to generate the mass at each node as follows:

$$\mathbf{M} = \sum_m \mathbf{M}_m \tag{6.10}$$

The damping matrix **D** is generally not assembled from the element damping matrices as is given in 3.19 because of the difficulty of determining the element damping parameter. Instead, it is usually calculated as a function of the global stiffness and mass matrices. In this thesis, the damping matrix is calculated as a linear function of the global mass matrix as follows:

$$\mathbf{D} = \frac{2\alpha}{\triangle t}\mathbf{M} \tag{6.11}$$

where α is a scaling factor. This ensures that the global damping matrix is diagonal which is needed for elemental level computations to avoid complex matrix inversion in explicit integration of the system dynamics.

Initialize Displacement Field and Apply load

Since a dynamic analysis is performed, initial conditions need to be taken into account. Also, as it can be seen in Eq. 3.27, the calculation of the displacement vector at the first time step ${}^{\Delta t}\mathbf{U}$ requires the displacement vectors from the previous two steps ${}^{0}\mathbf{U}$, ${}^{-\Delta t}\mathbf{U}$. The displacement variables are initialized assuming that the at time 0 the object is at rest which means that the nodal displacements vectors are zero at and prior to time step zero, i.e.

$$ {}^{-\Delta t}\mathbf{U} = \mathbf{0},\ {}^{0}\mathbf{U} = \mathbf{0} \tag{6.12} $$

The load is specified as a displacement constraint on the contact node where the load node displacement for the first time step is given by:

$$ {}^{\Delta t}\mathbf{U}_{load} = [u_{tool},\ v_{tool}] \tag{6.13} $$

where u_{tool} is the contact node displacement in the x direction, and v_{tool} is the contact node displacement in the y direction.

6.1.2 Time Stepping

Given all the matrices that have been calculated in the off-line precomputation stage above, in each time step the following calculations are performed in order to

find the 2D object response to the applied load:

- Compute the nodal elastic forces for each element as is given in Eq. 3.26 using elemental stiffness matrix and elemental nodal displacement vector that are known at time t

$$^{t}\mathbf{F}_{m} = \mathbf{K}_{m}.^{t}\mathbf{u}_{m} \tag{6.14}$$

- Since nodes are usually shared by a few elements, the contribution of each element should be summed in order to obtain the net nodal elastic forces at time t, $^{t}\mathbf{F}$:

$$\mathbf{F} = \sum_{m}^{t} \mathbf{F}^{m}$$

- Obtain the net nodal external forces vector at time t, ^{t}R which is zero at all the nodes of the mesh except the nodes at which external contact occurs. The nodes where an external forces exist and should be calculated include the load node and the boundary condition nodes. Equation 3.27 is used to calculate the external force on these nodes since the displacement at time $t + \triangle t$ at these nodes are known in Eq. 6.13. Therefore, for the load node the external force is

$$^{t}\mathbf{R}_{load} = \left(\frac{\mathbf{M}_{load}}{\triangle t^{2}} + \frac{\mathbf{D}_{load}}{2\triangle t}\right) {}^{t+\triangle t}\mathbf{U}_{load} + \left(\frac{\mathbf{M}_{load}}{\triangle t^{2}} - \frac{\mathbf{D}_{load}}{2\triangle t}\right) {}^{t-\triangle t}\mathbf{U}_{load} + \mathbf{F}_{load} - \left(\frac{2\mathbf{M}_{load}}{\triangle t^{2}}\right) {}^{t}\mathbf{U}_{load} \tag{6.15}$$

- Compute displacements using Eq. 3.27 that was derived based on the central difference method

- Update displacements to be used in the next time step:

$$^{t-\Delta t}\mathbf{U} = {}^{t}\mathbf{U} \quad , \quad {}^{t}\mathbf{U} = {}^{t+\Delta t}\mathbf{U} \tag{6.16}$$

and apply load for the next time step by increasing the displacement field on the load node which was specified by Eq. 6.13 above.

6.1.3 Simulation Results

To show the results of using the algorithm explained above to simulate the deformations in a soft object, the generated mesh shown in Fig. 6.2 is employed. Tabel 6.1 summarizes the parameters utilized in the linear analysis which are explained throughout the section.

Parameter	Value
Radius of the simulated body	0.1 m
Approximate Element edge length	0.02 m
Poisson's ratio (ν)	0.499
Young's modulus (E)	1800 Pa
Mass density (ρ)	1000 kg/m^3
Time step ($\triangle t$)	0.001 s
Damping scaling factor α	0.05

Table 6.1: Simulation parameters employed in linear FE analysis

The time step employed guarantees a stable system given the size of elements in the employed mesh. Based on the relationship given in Eq. 6.11 and the damping scaling factor utilized, each node in the mesh has a damping of 100 times its mass. In the simulations, Nodes 31, 42, and 43 are the fixed boundary condition nodes and the load is applied at Node 41. These nodes are specified by the user through the graphical user interface shown in Fig. 6.3.

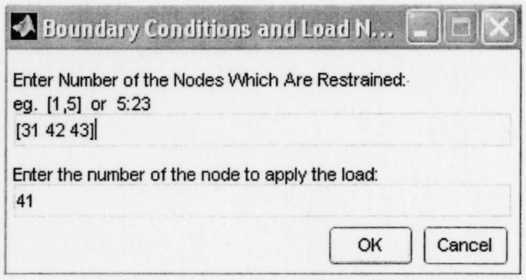


Figure 6.3: Graphical user interface to specify loading and boundary condition nodes

At each time step, the tool tip position (Node 41 position) is displaced by a loading step of 0 in the x axis direction and -0.0001 in the y axis direction. Fig. 6.4 shows a sequence of the deformed mesh at different stages. The first stage shows the mesh before deformation when no load is applied on the mesh. The second stage displays the deformed mesh after 200 time steps. The third figure is the deformed mesh after 400 time steps, and finally the last part is a comparison between the first and the third parts. The progress in the tool tip position over the time is shown as a think line.

To illustrate the linearity of the analysis, in Fig. 6.5, the deformations vs. external force profile in the y axis direction on Node 41 over 400 time steps for two different loading steps are compared. The first is for a loading step of -0.0001 in y axis direction and the second is for a loading step of -0.00005 in y axis direction. It can be seen that when the loading step is doubled, the external forces are doubled as well which is a property of a linear system.

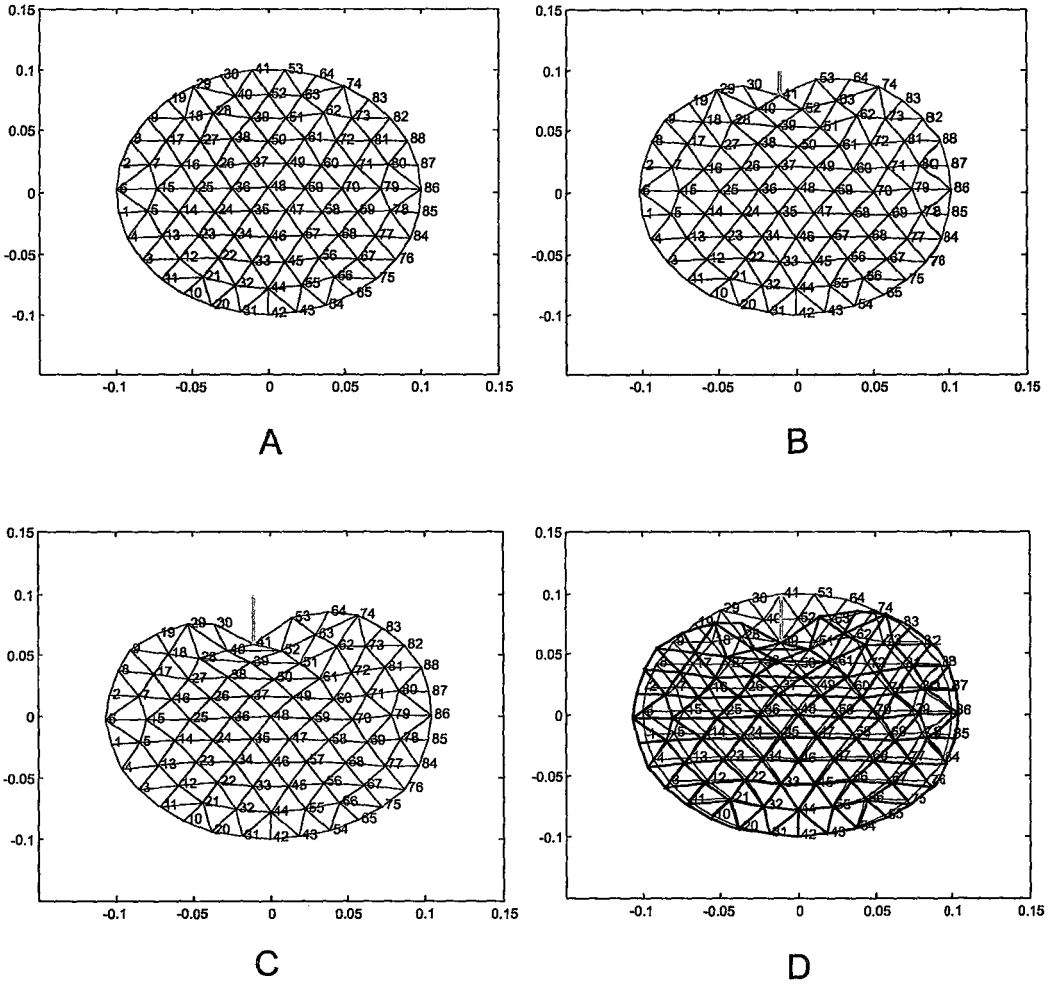


Figure 6.4: Simulation of soft-tissue deformation using linear analysis: (A) Original mesh. (B) Deformed mesh after 200 time steps. (C) Deformed mesh after 400 time steps. (D) Comparison between A and C.

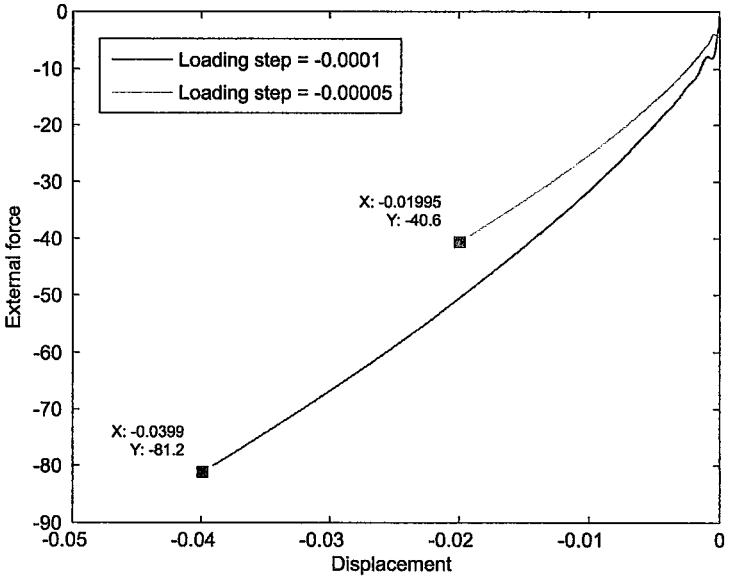


Figure 6.5: Displacement vs. external force profile at Node 41 in the Y axis direction for linear FE analysis

6.2 Steps and Results of 2D Explicit Dynamic Nonlinear Finite Element Algorithm

In general, the algorithm employed for the nonlinear analysis has the same steps as the linear analysis mentioned in the previous section and shown in Fig. 6.1. One of the main differences between the algorithms for the two types of analysis is calculation of the nodal elastic forces. In the nonlinear analysis, there is no need to calculate the elemental stiffness matrices since the elemental elastic force vectors are not calculated using the elemental stiffness matrices and elemental nodal displacement vectors as it given in Eq. 6.14. Instead the elastic forces are calculated using the following integral as it was given in Chapter 4:

$$\sum_{e} {}^{t}_{0}\mathbf{F}_m = \sum_{e} \int_{{}^{0}A_m} {}^{t}_{0}\mathbf{B}^{T}_{F_m} \, {}^{t}_{0}\hat{\mathbf{S}}_m \mathrm{d}^{0}A_m \tag{6.17}$$

where ${}^{t}_{0}\mathbf{B}^{T}_{F_m}$ is the elemental full strain-displacement matrix at at time t. Therefore, the spatial derivatives of the interpolation functions which are computed with respect to the initial configuration can be calculated off-line using Eq. 6.7 as it was explained in 6.1.1. The steps of computing the elemental nodal elastic forces for the nonlinear analysis are detailed below.

6.2.1 Elemental Nodal Elastic Force Vectors Calculation

To calculate the nodal elastic force vector for each element, the steps shown in Fig. 6.6 are performed for each element in the mesh. These steps are explained below.

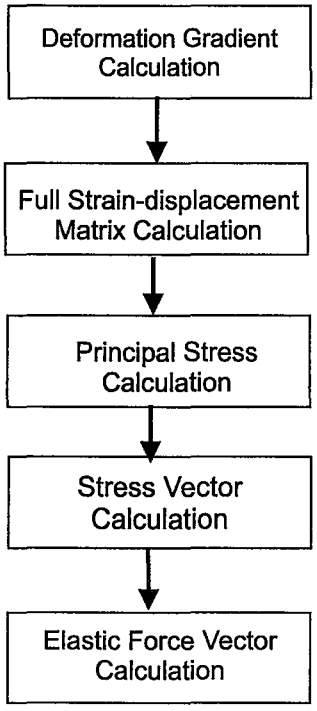


Figure 6.6: Steps to calculate nodal elastic force for each element

Calculation of the Deformation Gradient

The deformation gradient for a three node constant strain triangular element can be derived based on Eq. 4.10 and using the shape function given in Eq. 6.6. As it was given in Eq. 3.9, the shape function is employed to define the position of any point within the element in terms of the element nodes position as follows:

$$ {}^{t}\mathbf{p} = \sum_{k=1}^{3} h_k \, {}^{t}\mathbf{p}_k \tag{6.18} $$

where ${}^{t}\mathbf{p} = [{}^{t}x \; {}^{t}y]^T$ since a two-dimensional analysis is performed. Each component of the deformation gradient matrix can be calculated using the derivative of

the shape function with respect to the initial position of the nodes, i.e.

$$\frac{\partial^t p_i}{\partial^t p_j} = \sum_{k=1}^{3} \left(\frac{\partial h_k}{\partial^0 p_j} \right) {}^t p_i^k \tag{6.19}$$

Using the definition of the shape functions in Eq. 6.6, their derivatives with respect to the initial position of the nodes are

$$\begin{aligned} \frac{\partial h_1}{\partial^0 x} &= \frac{b_1}{2\triangle} \quad ; \quad \frac{\partial h_1}{\partial^0 y} = \frac{c_1}{2\triangle} \\ \frac{\partial h_2}{\partial^0 x} &= \frac{b_2}{2\triangle} \quad ; \quad \frac{\partial h_2}{\partial^0 y} = \frac{c_2}{2\triangle} \\ \frac{\partial h_3}{\partial^0 x} &= \frac{b_3}{2\triangle} \quad ; \quad \frac{\partial h_3}{\partial^0 y} = \frac{c_3}{2\triangle} \end{aligned} \tag{6.20}$$

where b_k and c_k are calculated as it was shown for Eq. 6.7 using the nodes positions in the initial configuration at time 0, and $\triangle$ is the area of each element in the initial cofiguration. Therefore, using Eq. 6.19 and Eq. 6.20, the deformation gradient matrix for each element can be written as:

$${}^t_0\mathbf{X} = \begin{bmatrix} \frac{\partial^t x}{\partial^0 x} & \frac{\partial^t x}{\partial^0 y} \\ \frac{\partial^t y}{\partial^0 x} & \frac{\partial^t y}{\partial^0 y} \end{bmatrix} = \frac{1}{2\triangle} \begin{bmatrix} b_1\,{}^t x_1 + b_2\,{}^t x_2 + b_3\,{}^t x_3 & c_1\,{}^t x_1 + c_2\,{}^t x_2 + c_3\,{}^t x_3 \\ b_1\,{}^t y_1 + b_2\,{}^t y_2 + b_3\,{}^t y_3 & c_1\,{}^t y_1 + c_2\,{}^t y_2 + c_3\,{}^t y_3 \end{bmatrix} \tag{6.21}$$

A different method for deriving the deformation gradient for a constant strain triangular element is given in [57] which would produce the same results as those obtained here.

Full Strain-displacement Matrix Calculation

The strain-displacement matrix in the nonlinear analysis is not constant as it was the case in the linear analysis in the previous section. The full strain-displacement matrix has to be recalculated for each element at every time step based on the constant strain-displacement matrix that is calculated off-line and the current deformation gradient as follows:

$$ {}^{t}_{0}\mathbf{B}_{F}^{(k)} = {}^{t}_{0}\mathbf{B}_{0}^{(k)} \, {}^{t}_{0}\mathbf{X}^{T} \tag{6.22} $$

where ${}^{t}_{0}\mathbf{B}^{(k)} \in \mathbb{R}^{3\times 2}$ are the two columns of the ${}^{t}_{0}\mathbf{B}$ that corresponds to node k and ${}^{t}_{0}\mathbf{B}_{F} \in \mathbb{R}^{3\times 6}$. The method for calculating ${}^{t}_{0}\mathbf{B}_{F}$ given in [29] produces the same results.

Second Piola-Kirchhoff Stress Vector Calculation

The second Piola-Kirchhoff Stress measure can be defined as symmetric matrix or a vector,

$$ {}^{t}_{0}\mathbf{S} = \begin{bmatrix} {}^{t}_{0}S_{11} & {}^{t}_{0}S_{12} \\ {}^{t}_{0}S_{21} & {}^{t}_{0}S_{22} \end{bmatrix} \quad ; \quad {}^{t}_{0}\hat{\mathbf{S}} = \begin{bmatrix} {}^{t}_{0}S_{11} \\ {}^{t}_{0}S_{22} \\ {}^{t}_{0}S_{12} \end{bmatrix} \tag{6.23} $$

The values of this measure are calculated for each element at every time step as follows:

1. Calculate the right Cauchy-Green deformation tensor ${}^{t}_{0}\mathbf{C}$ using the deformation gradient given in Eq. 4.14.

2. Perform the eigenvalue decomposition for the right Cauchy-Green deformation tensor in order to obtain the eigenvectors that represent the principal directions and the eigenvalues that represent the square of the principal stretches as explained in Sec. 4.2. The decomposition is given by:

$$ {}^{t}_{0}\mathbf{C} = \begin{bmatrix} Vp_1 & Vp_2 \end{bmatrix} \begin{bmatrix} \lambda_1^2 & 0 \\ 0 & \lambda_2^2 \end{bmatrix} \begin{bmatrix} Vp_1 & Vp_2 \end{bmatrix}^{-1} \tag{6.24} $$

where λ_1 and λ_2 are the principal stretches; Vp_1 and Vp_2 are the principal directions.

3. Use the principal stretches to calculate the principal values of the second Piola-Kirchhoff stress tensor according to Eq. 4.31 and Eq. 4.32. Many values for the bulk modulus κ where tested and it was found that a value higher than 10^4 results in a stable simulation while enforcing the incompressibility constrain. The higher the value of κ is the better the constrain is imposed; however, very large values could lead to very high forces generated through the calculations. Therefore, a value of 2×10^5 is employed during the simulations.

4. The second Piola-Kirchhoff stress tensor is calculated using its eigenvectors and eigenvalues. The principal stresses calculated in the previous step are the eigenvalues of the second Piola-Kirchhoff stress matrix. For isotropic hyperelastic materials, the principal axes for the stress tensor coincide with the principal axes of the strain tensor [75,76]. Therefore, the principal directions

of the second Piola-Kirchhoff stress tensor are the same as the principal directions of the right Cauchy-Green deformation tensor ${}^{t}_{0}\mathbf{C}$ that are obtained in Step 2 above. These eigenvectors are at the same time the principal directions of the right stretch matrix ${}^{t}_{0}\mathbf{U}$ as it was explained in Chapter 4. The second Piola-Kirchhoff stress vector that is used next is obtained by rearranging the components of the matrix and taking one of the off-diagonal components since the matrix is symmetric.

Elastic Force Vector Calculation

The elastic force vector for each element is computed based on the integral given in Eq. 6.17. The obtained full strain-displacement matrix in Eq. 6.22 and the second Piola-Kirchhoff stress vector are used to calculate the elastic force vector along with the area of each element as follows:

$$ {}^{t}_{0}\mathbf{F}_{m} = {}^{t}_{0}\mathbf{B}^{T}_{F_m} \, {}^{t}_{0}\hat{\mathbf{S}}_{m} \triangle \tag{6.25} $$

Finally, these elemental elastic force vectors are assembled in order to obtain the global force vector that is used to evaluate the displacements using Eq. 3.27 as explained in Sec. 6.1.

6.2.2 Simulation Results

Simulations were carried out using the same mesh and loading conditions as in the case of the linear analysis in Sec. 6.1.3. Also, the parameters employed in the non-linear analysis are the same as the ones summarized in Table 6.1 with the addition

of few other parameters summarized in Table 6.2.

Parameter	Value
Shear modulus μ	600 Pa [13]
Material constant α	-4.7 [13]
Bulk modulus κ	2×10^5 Pa

Table 6.2: Simulation parameters employed in nonlinear FE analysis

Fig. 6.7 shows a sequence of the deformed mesh at different stages by employing the algorithm for the nonlinear analysis. The first stage shows the original mesh. The second stage shows the deformed mesh after 200 time steps. The third figure is the deformed mesh after 400 time steps, and finally the last part is a comparison between the first and the third stages of the simulation. On the other side, Fig. 6.8 shows the state of the mesh when the load is released from the deformed configuration obtained earlier. It shows that the mesh relaxes back to the original undeformed configuration after few hundred time steps. The number of time steps required to get to the original configuration depends on the damping scaling factor employed.

The results obtained from the linear and nonlinear analysis are compared in Fig. 6.9 where the mesh deformations for the time step 400 are displayed together. It can be seen and it was also verified that the incompressibility constrain was enforced in the nonlinear analysis and the area of the mesh was kept almost constant before and after the deformations. The nonlinearity of the analysis is illustrated in Fig. 6.10 where the deformations vs. external force profile in the y axis direction on Node 41 over 400 time steps for two different loading steps (-0.0001 and -0.00005) are displayed. It can be seen that when the loading step is doubled, the external force is not just doubled which implies that the system demonstrates a nonlinear

behavior.

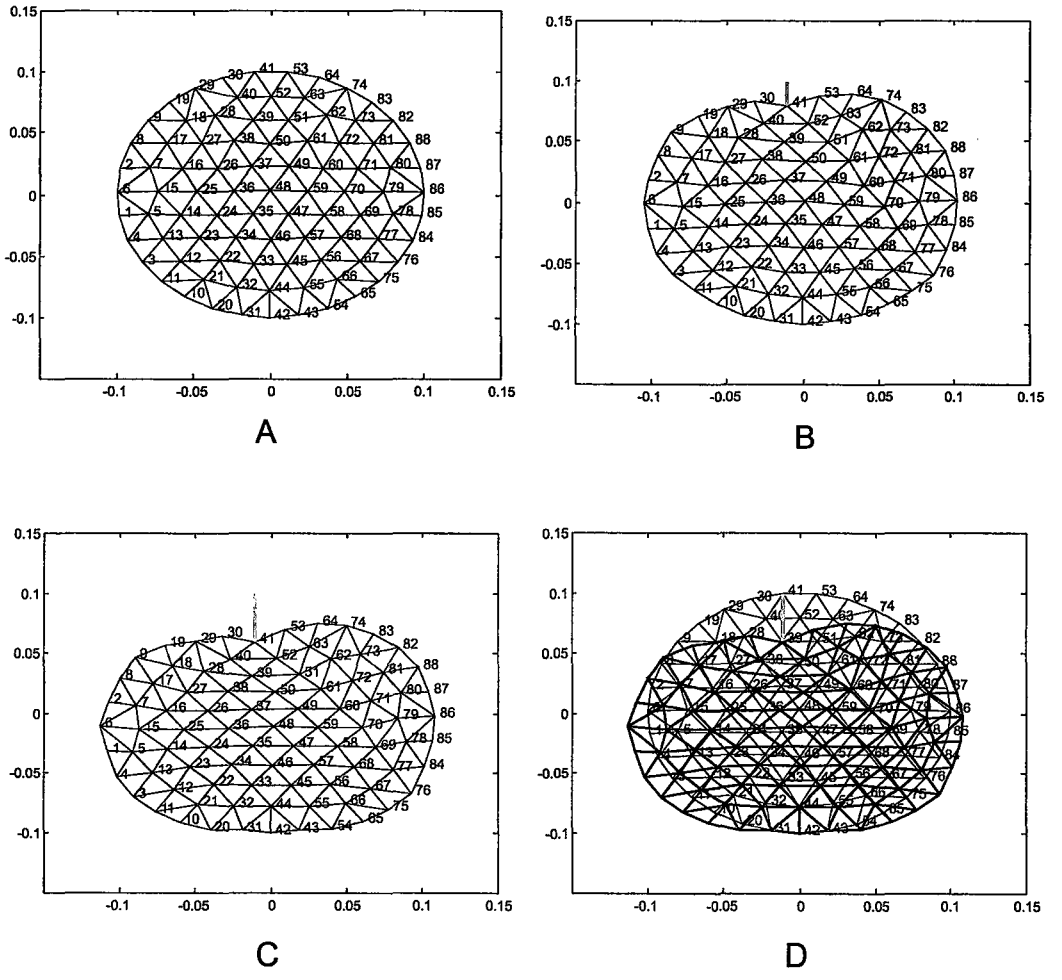


Figure 6.7: Simulation of soft-tissue deformation using nonlinear analysis: (A) Original mesh. (B) Deformed mesh after 200 time steps. (C) Deformed mesh after 400 time steps. (D) Comparison between A and C.

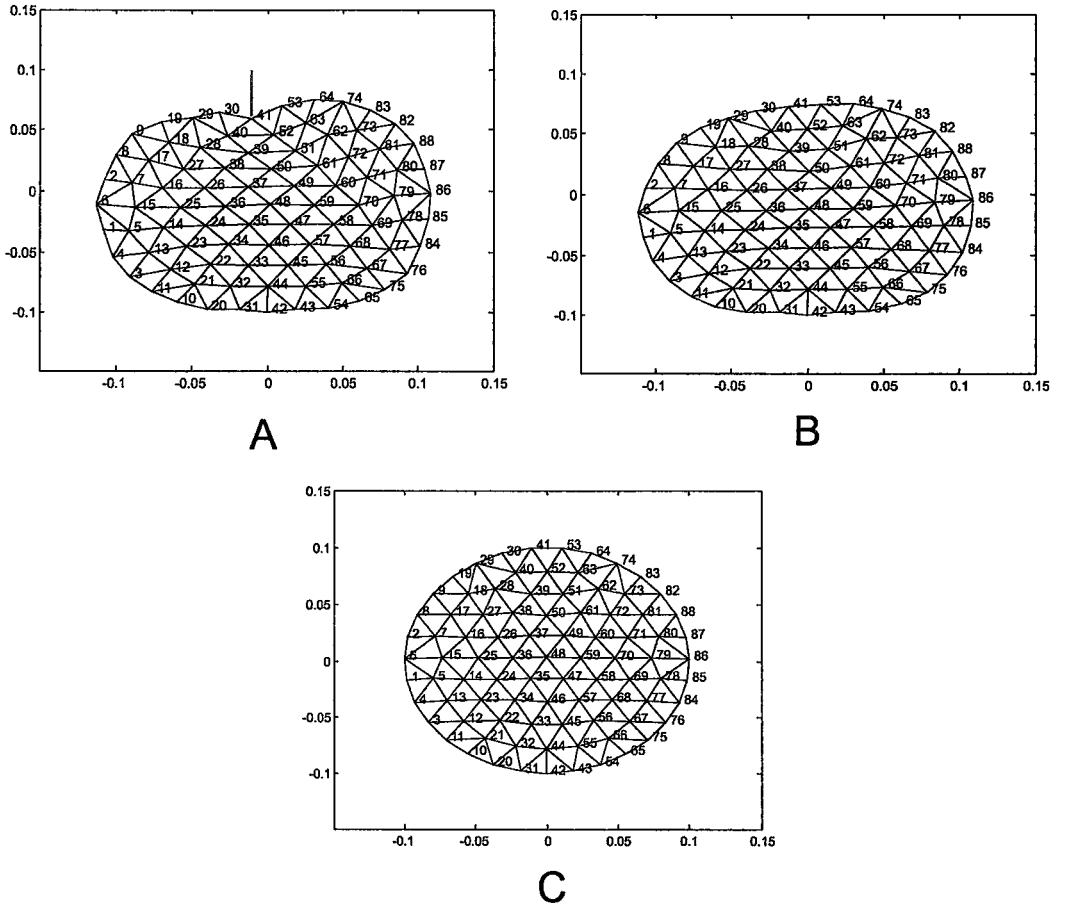


Figure 6.8: Simulation of soft-tissue relaxation from a deformed configuration using nonlinear analysis: (A) Deformed mesh. (B) Mesh configuration 50 time steps after removing the load. (C) Mesh relaxed to the original configuration.

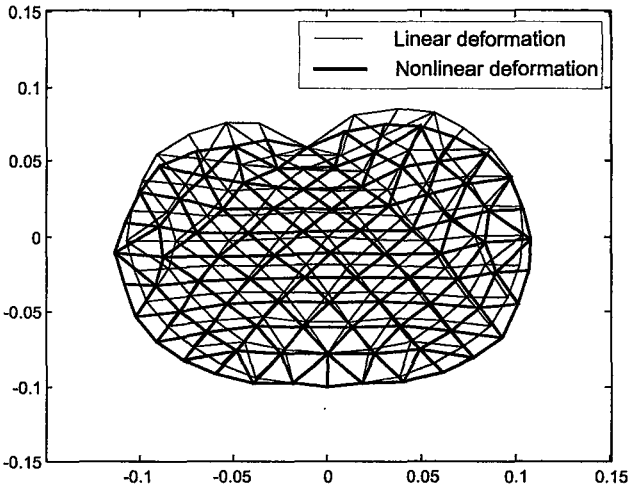


Figure 6.9: Comparison between the deformations obtained using linear and nonlinear model

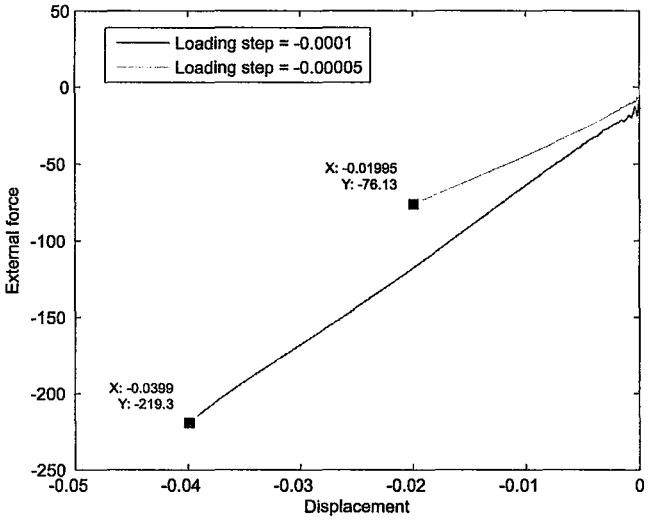


Figure 6.10: Displacement vs. external force profile on Node 41 in the Y axis direction for nonlinear FE analysis

6.3 Steps and Results of 2D Soft-tissue Cutting Simulations

The nonlinear analysis demonstrated in the previous section is employed to simulate cutting in soft-tissues. In order to let the simulated object deform prior to cutting and allow progressive cutting, the algorithm shown in Fig. 6.11 is developed. It can be seen that the types of the interactions between the tool tip and the simulated soft-tissue model are divided into three cases. The first case is free motion when there is no contact between the tool tip and the two-dimensional model. When there is contact between the tool tip and the object, the simulation is divided into two cases including deformation and cutting. A detailed explanation of the main blocks is given next.

6.3.1 Main Blocks in the Soft-tissue Cutting Algorithm

- **Off-line computations and initialization**: This step involves generating the mesh, getting boundary condition nodes from the user, calculating strain-displacement matrix, diagonal mass matrix and area for each element, assembling the global mass matrix, and initializing the displacement field. The details of these calculations have been explained earlier in the chapter.

- **Is there a collision?**: Since a simple two-dimensional circle is used to model the object, detecting a collision between the tool tip and the object is done by determining if the point that represents the tool tip is inside the radius of the circle or not. Obviously, more sophisticated collision detection routine should be employed for interaction with object of arbitrary shape.

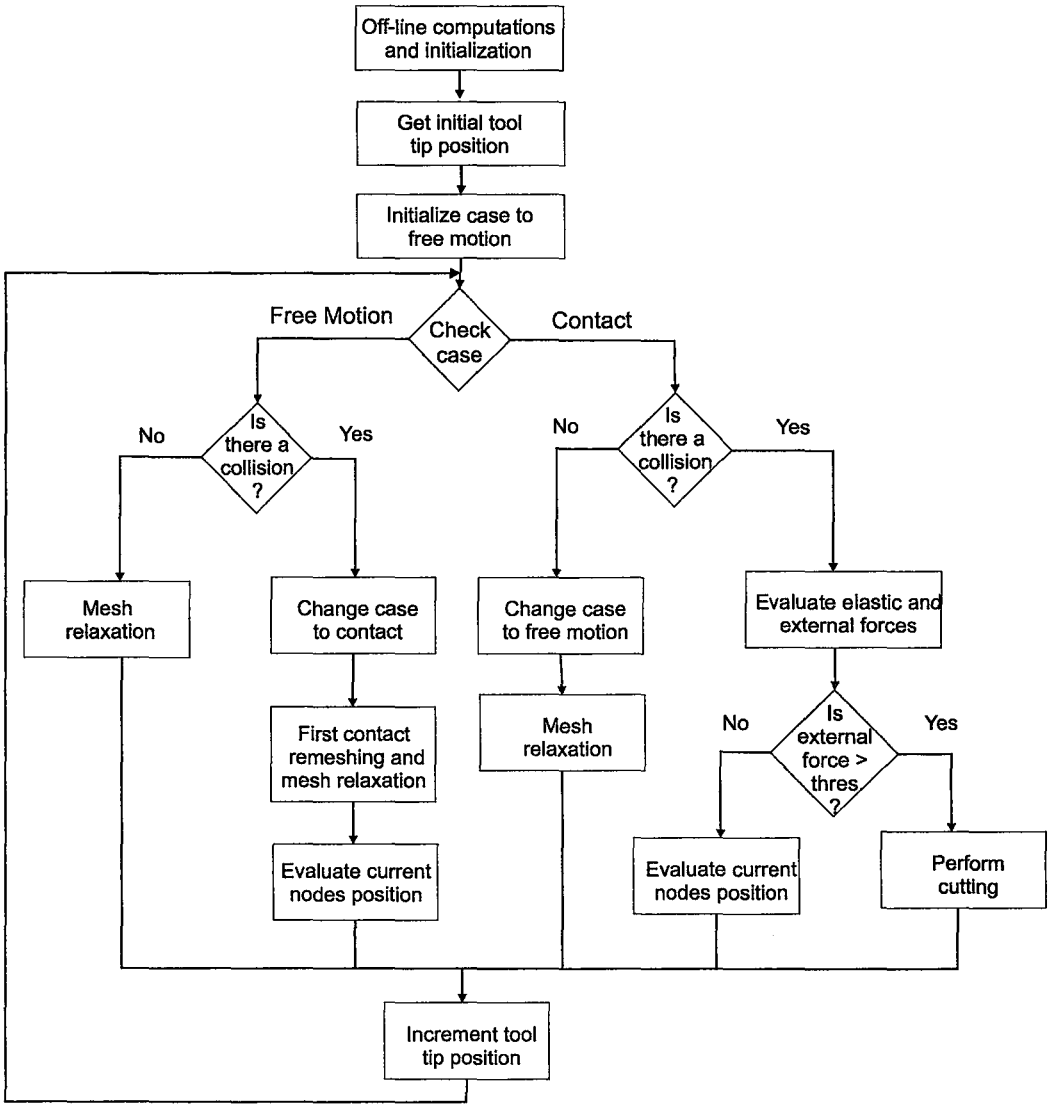


Figure 6.11: Main blocks of the cutting algorithm

- **First contact remeshing and mesh relaxation**: When the first contact between the tool tip and the object is detected, local remeshing should be performed by moving the closest boundary node to the contact point and applying the load on this node (load node). The strain-displacement matrix and area of the elements that contain the load node should be recalculated. After remeshing, few steps are performed in order to locally relax the mesh by calculating the deformation field of the nodes in the elements that contain the load node when the tool is just in contact with the object. A higher damping is used during these steps in order to minimize the oscillations and get to an equilibrium state faster. In order to prepare for the following time steps, the deformation field and current nodes position of the whole object are calculated when the load node is placed at the current tool tip position.

- **Mesh relaxation**: This step involves the calculations to evaluate the displacement field when there is no external load applied to the object. These calculations include the evaluation of the elastic force vector for nonlinear analysis as explained in Sec. 6.2.1 followed by the evaluation of the displacement field using Eq. 3.27. The position of each node at the current time step can be calculated by adding the obtained displacement field to the original position of each node.

- **Evaluate current nodes position**: To compute the position of the nodes in the mesh at the current time step, the same calculations performed for the mesh relaxation above should be repeated with the addition of computing the external force vector on the load node using Eq. 6.15. Therefore, this step computes the deformation in the object caused by the applied external load.

- **Is external force > threshold?**: To determine if a cutting procedure should be performed or not, the evaluated external force on the load node is decomposed into two components. The component in the tool direction is compared with a threshold value that represents the toughness of the soft-tissue. If the external force is less than the threshold value, the object is deformed and the deformation field is computed to update the nodes positions. On the other hand, if the external force exceeds the predefined threshold value, the cutting procedure is initiated.

- **Perform cutting**: The algorithm that was proposed in Chapter 5 is used to perform cutting.

6.3.2 Simulation Results of Soft-tissue Cutting

In order to illustrate the performance of the cutting algorithm explained above, a few scenarios will be shown bellow for different cases. In all these cases a constant threshold value of 40 N is assumed to be the maximum external force that can be applied to the soft-tissue before a cut occurs. As in the cases of linear and nonlinear analysis, a time step of 0.001 and α of 0.05 are utilized in the simulations. The initial position of the tool tip should be specified before the start of the simulation by specifying the x and y coordinates of the point which should be outside the mesh. The speed and direction of the load is also specified by choosing the loading (tool tip displacement) in the x and y directions for each time step. Using trial and error, it was found out that element quality of 0.2 guaranties numerical stability for a time step of 0.001. Therefore, the local remeshing algorithm must ensure that the quality of the elements are higher than 0.2. It should be noted that the tool

trajectory is assumed to be constant throughout the simulation which generates a straight line cut. By simple modifications to the algorithm employed, a curved cut can be easily generated.

Case 1: Cutting Simulation without Local Remeshing

Fig. 6.12 shows a sequence of steps when tool tip position starts at [0, 0.11] and is displaced by -0.0001 in the y axis direction at each time step. The sequence starts with the original undeformed mesh when the first contact between the object and the tool tip is detected. The second frame shows the node snapping of the closest boundary node (Node 53) to the contact point and assuming Node 53 to be the loading node. The third frame of Fig. 6.12 illustrates the deformed mesh before cutting occurs because the value of the external force on Node 53 in the tool direction is smaller than the chosen threshold value of 40N. When the value of the external force on Node 53 reaches the threshold value, the cutting procedure is performed by snapping the next internal node that makes the smallest angle with the cut path (Node 52) to the cut path and creating an a new node (Node 89) at the loading node (Node 51) in order to separate the elements as shown in the Part D of the figure. In the following time steps, the snapped node (Node 52) becomes the new loading node as shown in Part E until the threshold value is reached again. At that point the next internal node is snapped to the cut path as illustrated in Part F and a new node generated at Node 52.

The above process is repeated every time the external force crosses the threshold value. Fig. 6.13 shows the configuration of the mesh after 350 time step when few successive cut have been generated and the zoomed version of the mesh in

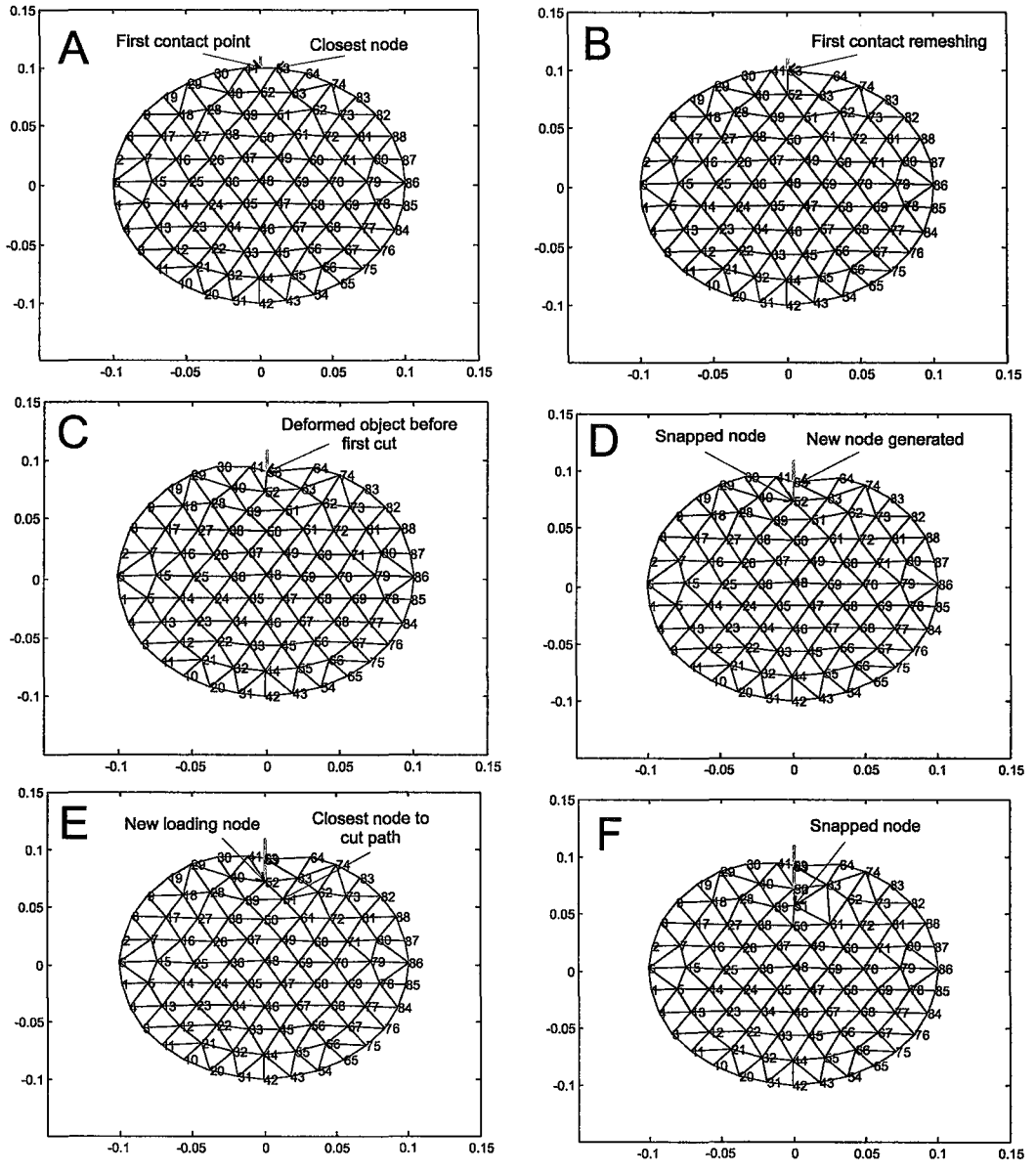


Figure 6.12: Case 1 simulation of soft-tissue cutting using nonlinear analysis: (A) Original mesh when first contact detected. (B) First contact remeshing. (C) Deformed mesh before the first cut. (D) Node snapping and elements separation. (E) New loading before second cut. (F) Node snapping and elements separation.

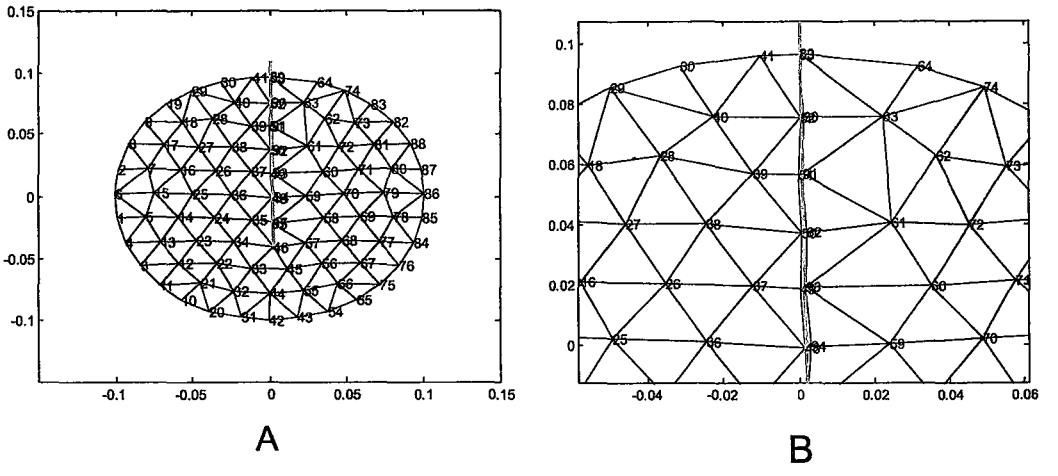


Figure 6.13: Case1 simulation of soft-tissue cutting after few successive cuts: (A) Mesh after 350 time steps. (B) A zoomed figure of part A to show separated elements on the cut path.

Part A illustrates the separated elements at the cut path with no overlapping between the elements at the two sides of the tool. It should be noted that in this case the local remeshing algorithm that is used to prevent the generation of small elements did not change the topology of the local elements or the position of the snapped node when cutting procedure was conducted. That is because all the generated elements after cut node snapping had good quality that did not require the remeshing procedure to be applied.

Fig. 6.14 shows the profile of the external force on the loading node. The component of the force vector along the tool direction that is used as the cutting criterion is shown in the figure. It can be seen that the force starts increasing after the first contact at time step 101. The external force keeps increasing until it passes the threshold value of 40 where a cutting procedure occurs and then the force drops in the following time step. After this drop, the force starts increasing again as

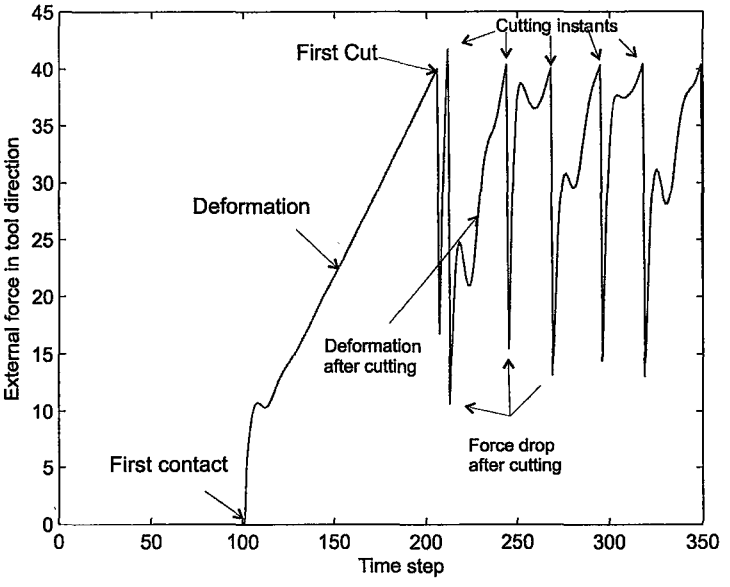


Figure 6.14: Case 1 force profile of the external force on the loading node along the tool direction over 350 time steps

loading continues on the next loading node until it reaches the threshold for the second time when the second cut occurs. This pattern is repeated as it can be observed very clearly in Fig. 6.14. The results of simulations and particularly the pattern of the force are consistent with the experimental results reported in the literature [1,66] (see Fig 5.8).

Case 2: Cutting Simulation with Local Retriangulation

To demonstrate the effectiveness of the algorithm developed in Sec. 5.2.2 which remeshes the local elements in the cutting area when badly shaped elements are generated after node snapping, the current cases and Case 3 are given as examples. In this case, the algorithm remeshes the local cutting area by only retriangulating

the elements using the Delaunay function in order to generate elements with good quality according to the criterion introduced in Eq. 5.1. The start position of the tool tip in this case is [0.08 0.07] and it is displaced by -0.0001 in the x axis direction at each time step.

Fig. 6.15 shows some of the important steps during the simulation. As in the previous case, the steps starts with the original undeformed mesh when the first contact between the object and the tool tip is detected. The second stage shows the node snapping of the closest boundary node (Node 83) to the contact point and assuming this node to be the loading node. The third part of Fig. 6.15 illustrates the deformed mesh before cutting occurs where the value of the external force on Node 83 in the tool direction is smaller than the threshold value. When the value of the external force on Node 83 reaches the threshold value, the cutting procedure is performed by snapping the next internal node that makes the smallest angle with the cut path (Node 73) to the cut path as shown in Part D. It can be seen that this node snapping generates a degenerated element that can cause numerical instability in the subsequent time steps. To avoid this, the proposed remeshing algorithm retriangulates the local elements and generates elements with better quality. The results of the retriangulation is shown in Part E of Fig. 6.15. It can also be observed that a new node (Node 89) has been created at the previous loading node (Node 83) in order to separate the elements, and that Node 73 becomes the loading node in the subsequent time steps until the next cut is initiated. The last stage in Part F displays the configuration of the mesh after 350 time steps when a few successive cuts have been performed.

Fig. 6.16 illustrates the remeshing procedure by showing a zoomed version of

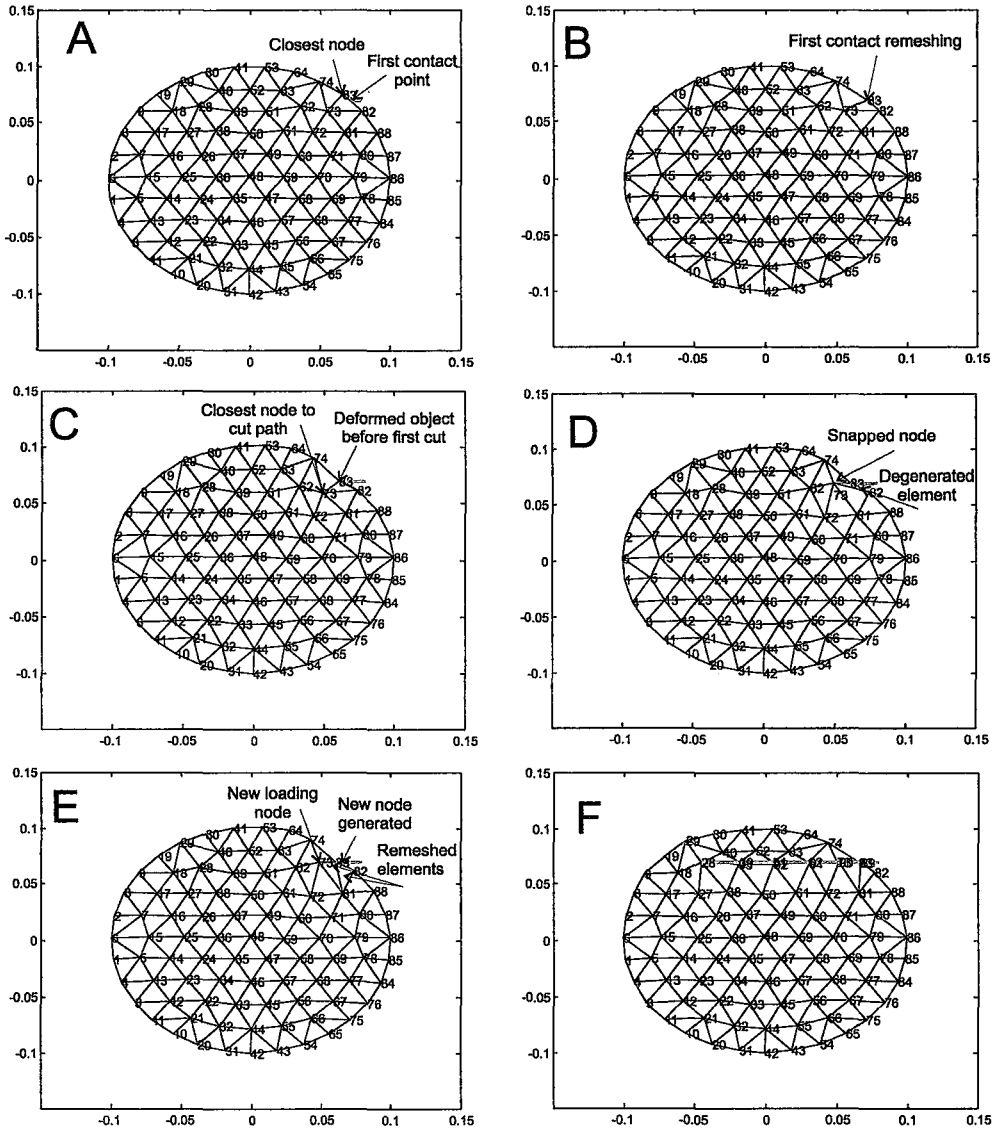


Figure 6.15: Case 2 simulation of soft-tissue cutting using nonlinear analysis: (A) Original mesh when first contact detected. (B) First contact remeshing. (C) Deformed mesh before the first cut. (D) Node snapping and degenerated element generation. (E) Local remeshing. (F) Successive cuts and mesh configuration after 350 time steps.

the mesh along with the local elements before and after the remeshing procedure. The profile of the external force in the tool direction on the loading node for this case is shown in Fig. 6.17.

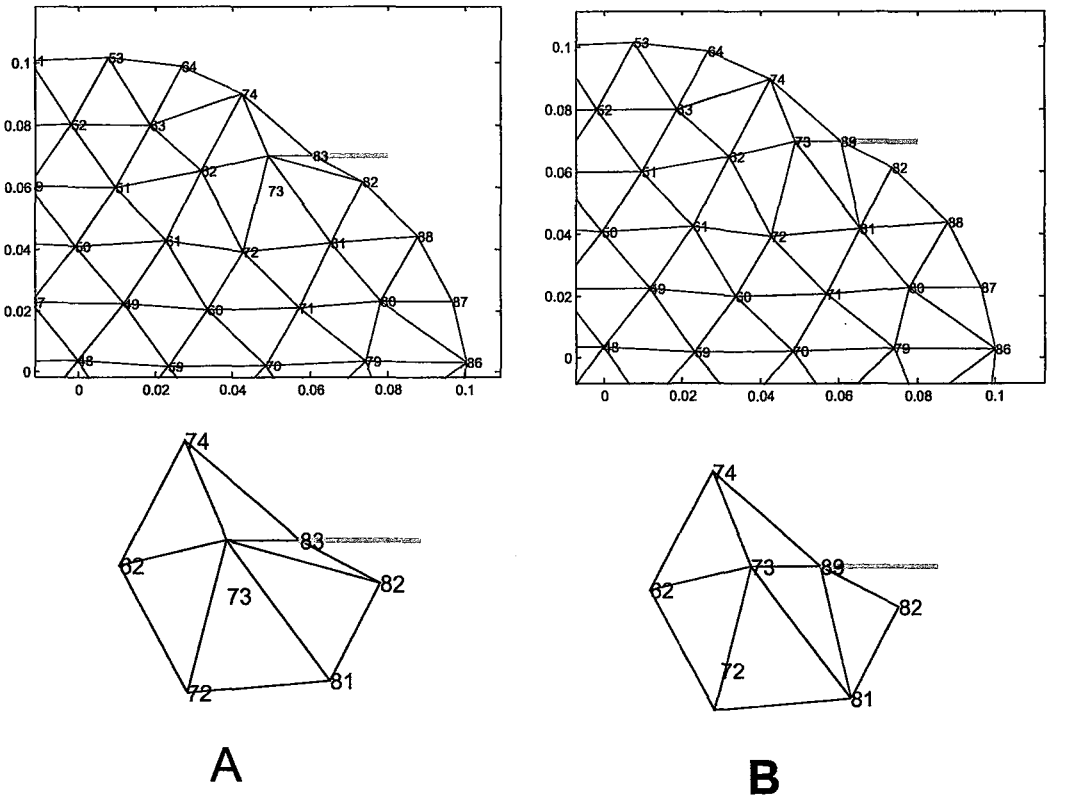


Figure 6.16: A zoomed version of parts D and E from Fig. 6.15 with local elements illustration: (A) Before remeshing. (B) After remeshing.

Case 3: Cutting Simulation with Local Retriangulation and Node Repositioning

In this case, the local remeshig algorithm requires the movement of the snapping node in order to generate elements with a quality higher than 0.2 because the deluanay function can not provide that. The start position of the tool tip is [0.095 0.05]

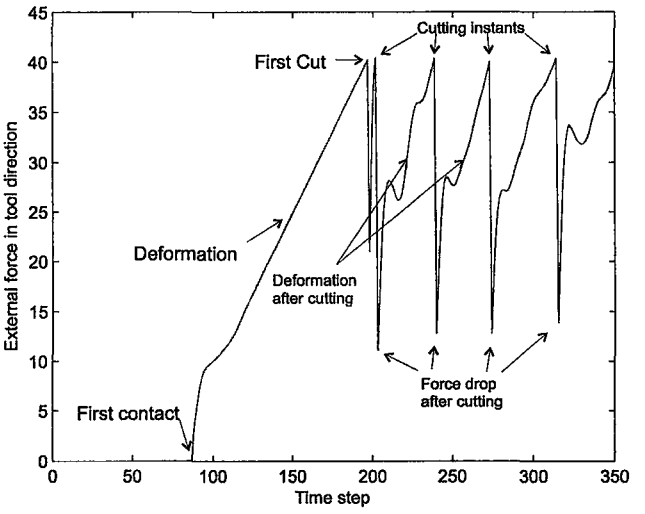


Figure 6.17: Case 2 force profile of the external force on the loading node in the tool direction over 350 time steps

and it is displaced by -0.0001 in the y axis direction at each time step.

Fig. 6.18 shows some of the important steps during the simulation. As in the previous case, the steps starts with the original undeformed mesh when the first contact between the object and the tool tip is detected. The second frame displays the node snapping of the closest boundary node (Node 87) to the contact point and assuming this node to be the loading node. The third part of Fig. 6.18 illustrates the deformed mesh before cutting occurs. When the value of the external force on Node 87 reaches the threshold value, the cutting procedure is performed by snapping the next internal node that makes the smallest angle with the cut path (Node 80) to the cut path as shown in Part D. It can be seen that this node snapping generates a degenerated element that can cause numerical instability in the following time steps. Therefore, the remeshing algorithm first retriangulates the

local elements; however, retriangulation does not generate elements with a quality higher than 0.2. As a result of this, the snapped node (Node 80) is moved by a resultant force dependent on the length of the edges connected to the node until the elements satisfy the quality measure as was explained in Sec 5.2.2.

Fig. 6.19 illustrates the remeshing procedure by showing the local elements at a sequence of steps until the elements have the required quality. Part E of Fig. 6.15 displays the mesh after the local remeshing and generating a new node (Node 89) at the previous loading node (Node 87) in order to separate the elements. Node 80 becomes the loading node in the following time steps until the next cut is preformed. The last frame shows the configuration of the mesh after 350 time steps when a few successive cuts have been performed. The profile of the external force in the tool direction on the loading node for this case is shown in Fig. 6.20.

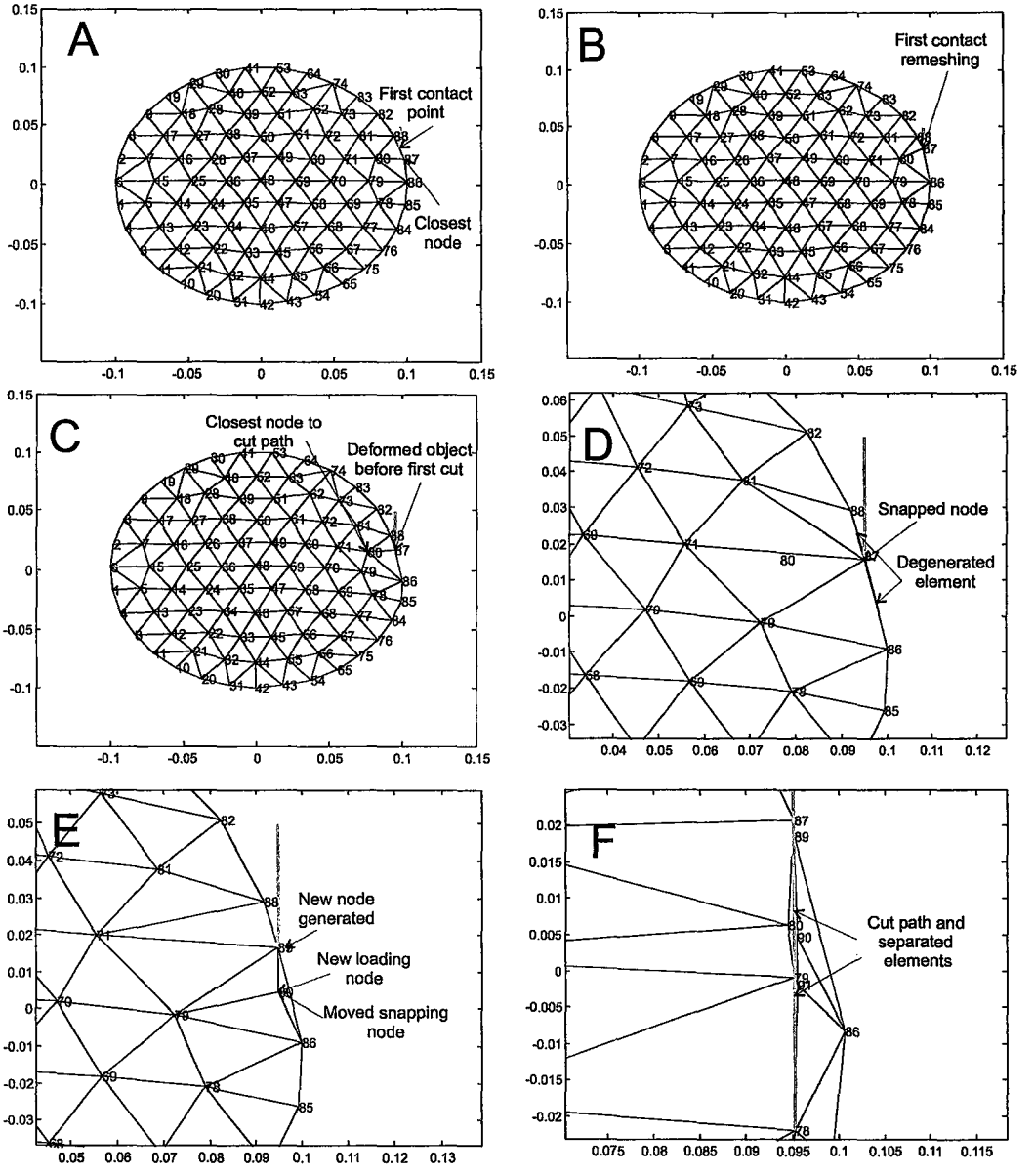


Figure 6.18: Case 3 simulation of soft-tissue cutting using nonlinear analysis: (A) Original mesh when first contact detected. (B) First contact remeshing. (C) Deformed mesh before the first cut. (D) Node snapping and degenerated element generation. (E) Local remeshing. (F) Successive cuts and mesh configuration after 350 time steps.

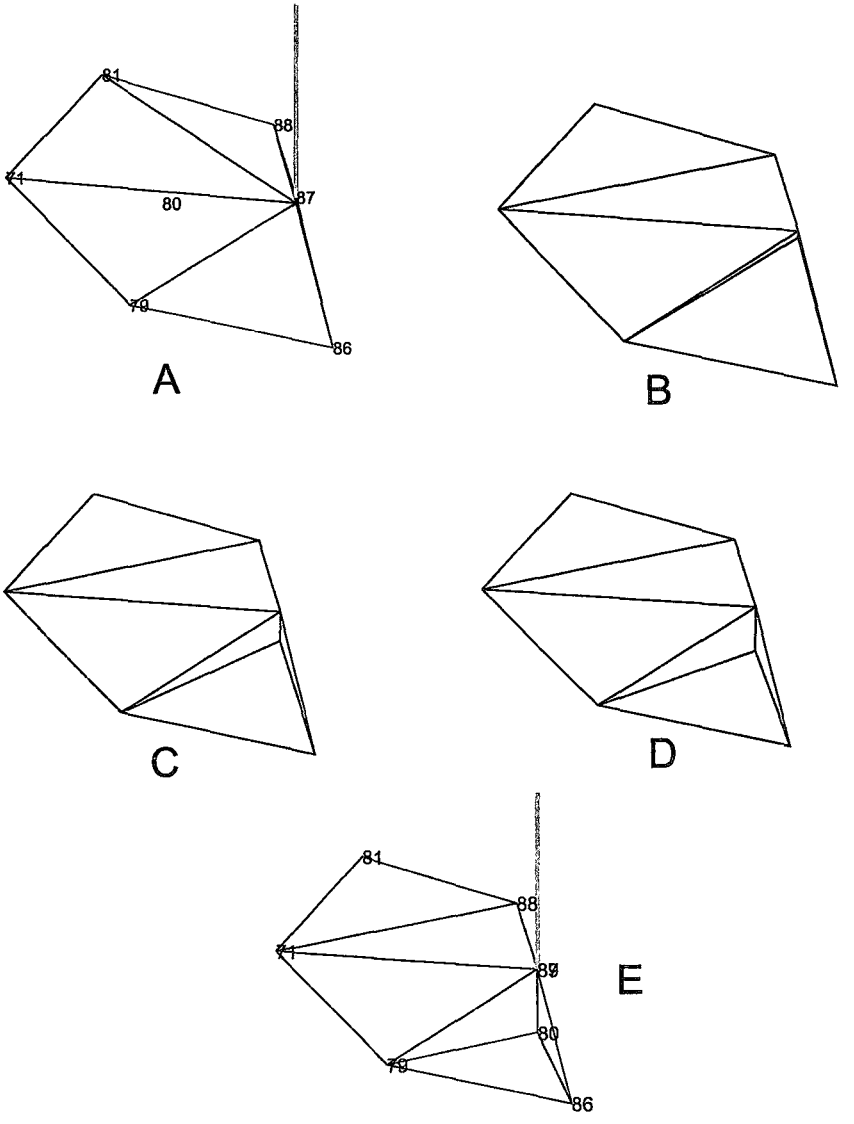


Figure 6.19: A zoomed version of the local remeshing from part D to E in Fig. 6.18 with local elements illustration: (A) Local elements before remeshing. (B,C,D) The remeshing procedure. (E) Local elements after remeshing.

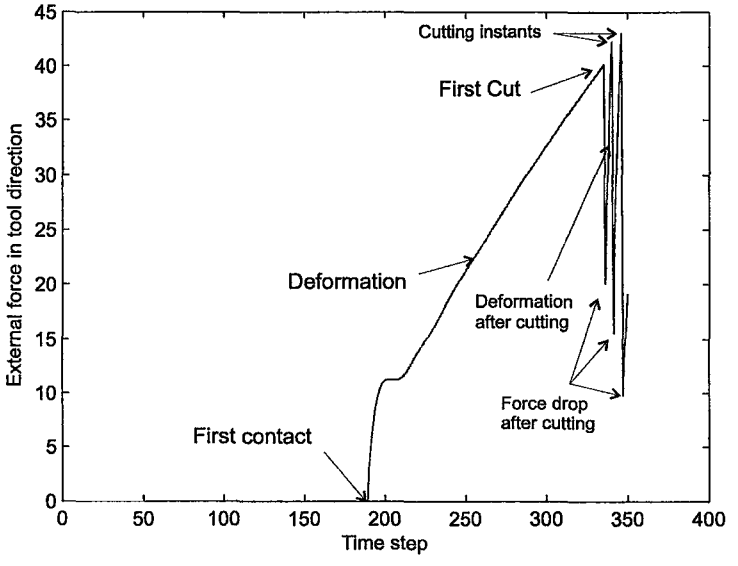


Figure 6.20: Case 3 force profile of the external force on the loading node in the tool direction over 350 time steps

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Virtual reality-based surgical simulators are emerging as a promising alternative to the conventional means of training. They can also be used for real-time pre-operative planning of surgical procedures (e.g. percutaneous therapy) as well as real-time (semi)-autonomous robotic execution of surgical tasks/procedures. These simulators provide numerous advantages to doctors and patients at the same time. They offer flexibility during training by allowing surgeons to practice on virtual patients as they would operate on real patients with realistic sensory feedback. The ability to adjust the tissue properties and operation scenarios in these simulators without being concerned about safety issues is critical for the initial stages of training. Also, they provide a number of other advantages including the ability to adjust the task difficulty level, quantitatively measure the trainee's progress, and provide active guidance. Many medical procedures involve some

form of instrument-soft tissue interaction and therefore, models of the human organs soft-tissue that can realistically reproduce haptic (force) and deformation response are needed. The complicated mechanical behavior of soft-tissues should be taken into account when modeling the deformations and interaction forces between surgical tool and tissues during the operation. There is consensus in the literature that linear models are insufficient for representing soft-tissue deformation response and that a nonlinear model should be employed for this purpose.

The contributions of this thesis in using nonlinear Finite Element analysis to model the mechanical behavior of tissues during tool-tissue interaction in two common surgical tasks, i.e. palpation and cutting. Geometrical and material nonlinearities are modeled. An incompressible nonlinear material is modeled using a nonlinear stress-strain measure based on a modified version of the Ogden hyperelastic constitutive equation. A penalty term has been added to the energy function to enforce the incomprehensibility of the tissue material. The total Lagrangian formulation with an explicit dynamic analysis is employed in the implementation.

An algorithm for the modeling of soft-tissue cutting procedures utilizing the nonlinear FE analysis with the Ogden-based constitutive equation has also been proposed. Element separation and node snapping is used to introduce the cut into the mesh. Node snapping can create degenerated elements with small size which can cause numerical instability in the simulation because of violating the time step criterion for explicit time integration and generating infinite deformations by zero area force calculation. Therefore, when such elements are generated after node snapping, they have to be treated in order to maintain the numerical stability of the simulation. In this thesis, this is achieved by remeshing the local elements

when degenerated elements are created. The remeshing process includes retriangulation of the elements using Delaunay function and/or moving the snapped node as needed in order to generate elements with the required quality.

7.2 Future Work

To enhance the simulation process and be able to develop a haptic-enabled surgery simulator that can incorporate the simulation of surgical prodding and cutting, some possibilities for future work on this project can include the following:

- Extending the simulation and modeling process to three-dimensional space which is closer to reality than the two-dimensional analysis implemented in this thesis. The general cutting algorithm developed can be applied to three-dimensional objects with simple modifications in the implementation of the algorithm, e.g. considering surfaces instead of lines.

- Enabling real-time simulation by having real-time deformation and force feedback rendering; haptics requires simulation rates in the range of a few hundred Hz whereas the graphics must be updated 30-60 Hz. The nonlinear analysis combined with a large number of nodes required for accurate modeling (several thousands nodes may be involved) can be computationally expensive. Meeting the computational requirement of such real-time simulations is beyond the power of existing conventional CPUs. Regardless of any improvement in the calculation speed, real-time calculation of deformations for meshes of such sizes requires significant computational power. The use of parallel processing, e.g. based on Field-Programmable Gate Array (FPGA)

or Graphics Processing Unit (GPU), can substantially improve the computation speed and hence enable real-time execution of nonlinear FE deformation models. The explicit dynamics algorithm that is used in this thesis has the advantage of element level computations which lend themselves rather neatly to parallelization.

- Incorporating time-dependent viscoelastic properties observed in the deformation of soft-tissue organs such as the brain into the modeling process [13, 23,33,35,77]. Such soft-tissues have elastic and viscus properties which make the strain rate dependent on time. Also, hysteresis is observed in the stress-strain curve of these viscoelastic soft-tissues. The Ogden-based viscoelastic energy function presented in [13,35] can be employed for modeling such tissues.

- Performing actual deformation experiments on soft-tissues to obtain the model parameters for different organs in order to have a simulated tissue response that matches that of the actual organ.

Bibliography

[1] T. Chanthasopeephan, *Characterization of Soft Tissue Cutting for Haptic Display: Experiments and Computational Models*. PhD thesis, Drexel University, Philadelphia PA, 2006.

[2] N. E. Seymour, A. G. Gallagher, S. A. Roman, M. K. OBrien, V. K. Bansal, D. K. Andersen, and R. M. Satava, "Virtual reality training improves operating room performance," *Ann Surg.*, vol. 236, pp. 458–464, October 2002.

[3] M. Bridges and D. Diamond, "The financial impact of teaching surgical residents in the operating room," *American Journal of Surgeons*, vol. 177, pp. 28–32, January 1999.

[4] R. M. Satava, "Virtual reality surgical simulator," *Surg Endosc*, vol. 7, pp. 203–205, May 1993.

[5] S. Suzuki, N. Suzuki, A. Hattori, M. Hayashibe, K. Konishi, Y. Kakeji, and M. Hashizume, "Tele-surgery simulation with a patient organ model for robotic surgery training," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 1, pp. 80 – 88, October 2005.

[6] M. Nakao, T. Kuroda, H. Oyama, G. Sakaguchi, and M. Komeda, "Physics-based simulation of surgical fields for preoperative strategic planning," *Journal of Medical Systems*, vol. 30, pp. 371–380, October 2006.

[7] M. Guidarelli, "Robotic surgery," *The Next Generation*, vol. 2, March 2006.

[8] W. H. Chapman, R. J. Albrecht, V. B. Kim, J. A. Young, and W. R. Chitwood, "Computer-assisted laparoscopic splenectomy with the da vinci^{TM} surgical robot," *Journal of Laparoendoscopic and Advanced Surgical Techniques*, vol. 12, pp. 155–159, June 2002.

[9] A. R. Lanfranco, A. E. Castellanos, J. P. Desai, and W. C. Meyers, "Robotic surgery: A current perspective," *Annals of Surgery*, vol. 239, pp. 14–21, January 2004.

[10] A. A. Haggag, "Robotic surgery: When technology meets surgical precision," *The Internet Journal of Health*, vol. 5, no. 1, 2006.

[11] Y. Zhuang, *Real-time simulation of physically realistic global deformations*. Ph.d. thesis, University of California, Berkeley,CA, 2000.

[12] K. Miller, K. Chinzei, G. Orssengo, and P. Bednarz, "Mechanical properties of brain tissue in-vivo: experiment and computer simulation," *Journal of Biomechanics*, vol. 33, pp. 1369–1376, November 2000.

[13] K. Miller, Z. Taylor, and A. Wittek, "Mathematical models of brain deformation behavior for computer-integrated neurosurgery," Research Report ISML/01/2006, The University of Western Australia, Intelligent Systems for Medicine Laboratory, 2006.

[14] K. Miller, G. Joldes, D. Lance, and A. Wittek, "Total lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation," *Communications in numerical methods in engineering*, vol. 23, no. 2, pp. 121–134, 2007.

[15] H. Delingette and N. Ayache, "Hepatic surgery simulation," *Communications of the ACM*, vol. 48, pp. 31 – 36, February 2005.

[16] P. Wang, A. Becker, I. Jones, A. Glover, S. Benford, C. Greenhalgh, and M. Vloeberghs, "A virtual reality surgery simulation of cutting and retraction in neurosurgery with force-feedback," *Computer Methods and Programs in Biomedicine*, vol. 84, pp. 11–18, October 2006.

[17] P. Wang, A. Becker, I. Jones, A. Glover, S. Benford, C. Greenhalgh, and M. Vloeberghs, "Virtual reality simulation of surgery with haptic feedback based on the boundary element method," *Computers and Structures*, vol. 85, pp. 331–339, April 2007.

[18] K. Hansen, L. Brix, C. Pedersen, J. Haase, and O. Larsen, "Modelling of interaction between a spatula and a human brain," *Medical Image Analysis*, vol. 8, pp. 23–33, March 2004.

[19] W. Chou and T. Wang, "Human-computer interactive simulation for the training of minimally invasive neurosurgery," in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, pp. 1110–1115, Oct 2003.

[20] P. Lamata, E. Gmez, F. Snchez-Margallo, . Lpez, C. Monserrat, V. Garca, C. Alberola, M. R. Florido, J. Ruiz, and J. Usn, "Sinergia laparoscopic virtual reality simulator: Didactic design and technical development," *Computer Methods*

and Programs in Biomedicine, vol. 85, pp. 273–283, March 2007.

[21] K. Salisbury, F. Conti, and F. Barbagli, "Haptic rendering: introductory concepts," *Computer Graphics and Applications,IEEE*, vol. 24, pp. 24–32, March/April 2004.

[22] C. Basdogan, S. De, J. Kim, M. Muniyandi, H. Kim, and M. A. Srinivasan, "Haptics in minimally invasive surgical simulation and training," *Computer Graphics and Applications,IEEE*, vol. 24, pp. 56–64, March/April 2004.

[23] H. Delingette, "Toward realistic soft-tissue modeling in medical simulation," *Proceedings of the IEEE*, vol. 86, pp. 512–523, March 1998.

[24] C. Mendoza and C. Laugier, "Tissue cutting using finite elements and force feedback," in *Surgery Simulation and Soft Tissue Modeling* (N. Ayache and H. Delingette, eds.), vol. 2673 of *Lecture Notes in Computer Science*, pp. 1003–1004, Springer Berlin / Heidelberg, January 2003.

[25] S. Cotin, H. Delingette, and N. Ayache, "A hybrid elastic model for real-time cutting, deformations, and force feedback for surgery training and simulation," *The Visual Computer*, vol. 16, pp. 437–452, 2000.

[26] A. Frank, I. Twombly, T. Barth, and J. Smith, "Finite element methods for real-time haptic feedback of soft-tissue models in virtual reality simulators," in *Proceedings of the Virtual Reality 2001 Conference (VR'01)*, pp. 257–263, March 2001.

[27] J. Berkley, G. Turkiyyah, D. Berg, M. Ganter, and S. Weghorst, "Real-time finite element modeling for surgery simulation: an application to virtual suturing,"

IEEE Transactions on Visualization and Computer Graphics, vol. 10, pp. 314–325, May-June 2004.

[28] D. V. G. Ian M. Smith, *Programming the Finite Element Method: With Application to Geomechanics*. John Wiley and Sons, 2004.

[29] K.-J. Bathe, *Finite Element Procedures*. Prentice-Hall, 1996.

[30] M. Bro-Nielsen, "Finite element modeling in surgery simulation," *Proceedings of the IEEE*, vol. 86, pp. 490–503, March 1998.

[31] X. Wu, M. Downes, T. Goktekin, and F. Tendick, "Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes," *Computer Graphics Forum*, vol. 20, pp. 349–358, September 2001.

[32] G. Picinbono, H. Delingette, and N. Ayache, "Non-linear anisotropic elasticity for real-time surgery simulation," *Graphical Models*, vol. 65, pp. 305–321, September 2003.

[33] Y. Tillier, A. Paccinia, J. Delottea, M. Durand-Revillea, and J.-L. Chenota, "Finite element modelling for soft tissues surgery based on nonlinear elasticity behaviour," *International Congress Series*, vol. 1268, pp. 384–389, June 2004.

[34] K. Miller and K. Chinzei, "Constitutive modelling of brain tissue: experiment and theory," *Journal of Biomechanics*, vol. 30, pp. 1115–1121, November 1997.

[35] K. Miller and K. Chinzei, "Mechanical properties of brain tissue in tension," *Journal of Biomechanics*, vol. 35, pp. 483–490, April 2002.

[36] A. Wittek, J. Laporte, and K. Miller, "omputing reaction forces on surgical tools for robotic neurosurgery and surgical simulation," in *Proceedings of the Australasian Conference on Robotics and Automation*, 2004.

[37] C. G. F. T. W. Nowinski, "A finite element method based deformable brain atlas suited for surgery simulation," in *Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, (Shanghai, China), pp. 4337–4340, September 2005.

[38] A. Horton, A. Wittek, and K. Miller, "Towards meshless methods for surgical simulation," in *Proceedings of MICCAI Workshop*, (Copenhagen), pp. 34–43, Oct 2006.

[39] S. J. Hollister, "Constitutive equations: Linear and nonlinear elasticity." class notes for BME/ME 456, Department of Biomedical Engineering, University of Michigan.

[40] S. J. Hollister, "Large deformation nonlinear mechanics: Stress, strain and constitutive defintions." class notes for BME/ME 506, Department of Biomedical Engineering, University of Michigan.

[41] ANSYS, Inc., *Theory Reference*.

[42] R. W. Ogden, "Large deformation isotropic elasticity: On the correlation of theory and experiment for compressible rubberlike solids," *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 328, pp. 567–583, June 1972.

[43] R. S. Rivlin, G. I. Barenblatt, and D. D. Joseph, *Collected Papers of R.S. Rivlin.* Springer, 1997.

[44] R. D. W. Javier Bonet, *Nonlinear Continuum Mechanics for Finite Element Analysis.* Cambridge University Press, 1997.

[45] O. H. Varga. Interscience Publishers, 1966.

[46] T. R. T Veerarajan, *Numerical Methods with Programs in C.* Tata McGraw-Hill, 2006.

[47] C. Basdogan and M. A. Srinivasan, "Haptic rendering in virtual environments," in *Handbook of Virtual Environments: Design, Implementation, and Applications* (K. M. Stanney, ed.), (Mahwah, NJ), p. 117, Lawrence Erlbaum Associates, 2002.

[48] A. Nealen, M. Mller, R. Keiser, E. Boxerman, and M. Carlson, "Physically based deformable models in computer graphics," in *EUROGRAPHICS*, State of The Art Report (STAR), 2005.

[49] C. Mendoza, *Soft Tissue Interactive Simulations for Medical Applications Including 3D Cutting and Force Feedback.* PhD thesis, INRIA Rhone-Alpes, INPG, Grenoble, France, 2003.

[50] S. Gibson and B. Mirtich, "A survey of deformable modeling in computer graphics," Technical Report TR-97-19, Mitsubishi Electric Research Lab., Cambridge, MA, November 1997.

[51] H. akmak and U. Khnapfel, "Animation and simulation techniques for vr-training systems in endoscopic surgery," in *Computer Animation and Simulation*

2000 (B. A. Nadia Magnenat-Thalmann, Daniel Thalmann, ed.), Eurographics Workshop, pp. 173–185, Springer, 2000.

[52] J. Brown, S. Sorkina, J.-C. Latombea, K. Montgomeryb, and M. Stephanidesb, "Algorithmic tools for real-time microsurgery simulation," *Medical Image Analysis*, vol. 6, pp. 289–300, September 2002.

[53] D. James and D. Pai, "Accurate real time deformable objects," in *Proccedings of Siggraph*, pp. 65–72, August 1999.

[54] P. K. Banerjee, *The Boundary Element Methods in Engineering*. McGraw-Hill College, 1994.

[55] A. Oram and G. Wilson, *Beautiful Code: Leading Programmers Explain How They Think*. O'Reilly, 2007.

[56] R. T. O.C. Zienkiewicz, *The finite element method*, vol. 1. Butterworth-Heinemann, 5 ed., 2000.

[57] A. Munjiza, *The Combined Finite-discrete Element Method*. John Wiley and Sons, 2004.

[58] M. Bro-Nielsen and S. Cotin, "Real-time volumetric deformable models for surgery simulation using finite elements and condensation," in *EUROGRAPHICS* (J. Rossignac and F. Sillion, eds.), vol. 15, Blackwell, 1996.

[59] A. B. Mor, *Progressive Cutting with Minimal New Element Creation of Soft Tissue Models for Interactive Surgical Simulation*. PhD thesis, The Robotics Institute,Carnegie Mellon University, Pittsburgh, PA, October 2001.

[60] S. De, J. Kim, and M. Srinivasan, "A meshless numerical technique for physically based real time medical simulations," in *Medicine Meets Virtual Reality 2001* (J. Westwood, H. M. Hoffman, G. T. Mogel, D. Stredney, and R. A. Robb, eds.), pp. 113–118, IOS Press, 2001.

[61] H. W. Nienhuys and A. F. van der Stappen, "A surgery simulation supporting cuts and finite element deformation," in *Medical Image Computing and Computer Assisted Intervention (MICCAI)* (W. Niessen and M. Viergever, eds.), vol. 2208 of *LNCS*, pp. 153–160, Springer-Verlag, October 2001.

[62] D. Serby, M. Harders, and G. Szkely, "A new approach to cutting into finite element models," in *Proceedings of the 4th International Conference on Medical Image Computing and Computer-Assisted Intervention* (W. Niessen and M. Viergever, eds.), vol. 2208 of *Lecture Notes In Computer Science*, pp. 425 – 433, Springer-Verlag, 2001.

[63] D., M. Harders, M. Gross, and G. Szekely, "Hybrid cutting of deformable solids," in *Proceedings of the IEEE Virtual Virtual Reality Conference*, pp. 35–42, March 2006.

[64] C. Bruyns, S. Senger, A. Menon, K. Montgomery, S. Wildermuth, and R. Boyle, "A survey of interactive mesh-cutting techniques and a new method for implementing generalized interactive mesh cutting using virtual tools," *The Journal of Visualization and Computer Animation*, vol. 13, pp. 21–42, February 2002.

[65] H.-W. Nienhuys, *Cutting in deformable objects*. Ph.d. thesis, Utrecht University, 2003.

[66] Y.-J. Lim, W. Jin, and S. De, "On some recent advances in multimodal surgery simulation: A hybrid approach to surgical cutting and the use of video images for enhanced realism," *Presence*, vol. 16, pp. 563–583, December 2007.

[67] D. Bielser and M. Gross, "Interactive simulation of surgical cuts," in *Proceedings of the 8th Pacific Conference on Computer Graphics and Applications*, PG, pp. 116–125, IEEE Computer Society, October 2000.

[68] D. Bielser, P. Glardon, M. Teschner, and M. Gross, "A state machine for real-time cutting of tetrahedral meshes," *Graphical Models*, vol. 66, pp. 398–417, 2004.

[69] T. K. Huynh Quang Huy Viet and H. T. Tanaka, "An adaptive 3d surface mesh cutting operation," in *AMDO* (F. Perales and R. Fisher, eds.), vol. 4069 of *LNCS*, pp. 366–374, 2006.

[70] D. Zhou, M. R. Claffee, K.-M. Lee, and G. V. McMurray, "Cutting, by pressing and slicing, applied to the robotic cut of bio-materials, part ii: Force during slicing and pressing cuts," in *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, (Orlando, Florida), pp. 2256–2261, May 2006.

[71] P.-O. Persson and G. Strang, "A simple mesh generator in matlab," *SIAM Review*, vol. 46, pp. 329–345, June 2004.

[72] D. A. Field, "Qualitative measures for initial meshes," *International Journal for Numerical Methods in Engineering*, vol. 47, pp. 887 – 906, Jan 2000.

[73] N.-S. Lee and K.-J. Bathe, "Error indicators and adaptive remeshing in large deformation finite element analysis," *Finite Elements in Analysis and Design*, vol. 16, pp. 99–139, May 1994.

[74] W. N. W. R. H. Crawford, D. C. Anderson, "Mesh rezoning of 2d isoparametric elements by inversion," *International Journal for Numerical Methods in Engineering*, vol. 28, no. 3, pp. 523–531, 1989.

[75] J. P. Ward, *Solid Mechanics: An Introduction*. Springer, 1992.

[76] M. Jirasek and Z. P. Bazant, *Inelastic analysis of structures*. John Wiley and Sons, 2001.

[77] Y. Fung, *Biomechanics: Mechanical Properties of Living Tissues*. Springer-Verlag, second ed., 1993.