# Ultrasound Image and 3D Finite Element based Tissue Deformation Simulator for Prostate Brachytherapy 

by

Orcun Goksel
B.S., Middle East Technical University, Ankara, TURKEY, 2002

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF APPLIED SCIENCE
in
THE FACULTY OF GRADUATE STUDIES
(Electrical and Computer Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA
December 2004
© Orcun Goksel, 2004

## Abstract

Brachytherapy is a prostate cancer treatment where radioactive pellets (seeds) are implanted into prostate region through long flexible needles. Steering them to preplanned locations require the use of tip bevel deflection effect and a thorough comprehension of tissue deformation and needle flexibility. A simulator is proposed to help medical residents in acquiring these procedural skills. In prostate brachytherapy, two important feedback modalities during needle insertions are the transrectal ultrasound (TRUS) imaging and the kinesthetic feedback received through the needle base. Thus, these have been incorporated in the simulator, the subsystems of which are presented here.

In this thesis, a needle-tissue interaction model from prior work which was based on finite element method is extended into 3D and adapted for custom meshes. The issues related to needle-tissue coupling are addressed. A meshing scheme which generates soft tissue models with small number of nodes for haptic implementation from contours on parallel slices is proposed. It is utilized to generate a prostate volume mesh which consists of well-conditioned elements for finite element analysis. A simulation interface is developed where a 3D rendering of the model deformation is displayed while the needle can be manipulated by entering target positions for its base.

A method for B-mode TRUS image synthesis out of a deformed tissue volume for which the patient data was collected at the BC Cancer Agency is also presented. Some interpolation schemes for remotely spaced ultrasound data are compared on a sample case and a method for overlaying needle shaft reflection onto TRUS images is described.

## Table of Contents

Abstract ..... ii
Table of Contents ..... iii
List of Tables ..... vi
List of Figures ..... vii
Glossary ..... ix
Acknowledgements ..... xi
1 Introduction ..... 1
1.1 Prostate Cancer ..... 1
1.2 Brachytherapy ..... 2
1.3 Need for Training ..... 5
1.4 Related Work ..... 7
1.4.1 Tissue Modeling ..... 8
1.4.2 Needle Insertion ..... 10
1.4.3 Meshing Techniques ..... 11
1.4.4 Ultrasound Imaging and 3D Reconstruction ..... 13
1.5 Proposed Simulator ..... 16
1.6 Thesis Organization ..... 16
2 Needle-Tissue Interaction Model ..... 17
2.1 Stiffness Matrix Inversion and Condensation ..... 17
2.1.1 Inversion ..... 18
2.1.2 Condensed Subsystem ..... 19
2.2 Contact Behaviour ..... 19
2.3 Boundary Condition Change ..... 21
2.4 Local Frame Rotation ..... 22
2.4.1 Stick State ..... 23
2.4.2 Slip State ..... 24
2.5 Decision of Remeshing ..... 27
2.6 Mesh Modification ..... 29
2.6.1 $\mathcal{O}\left(m n^{2}\right)$ Approach ..... 30
2.6.2 $\mathcal{O}\left(m^{2} n\right)$ Approach ..... 31
2.7 Needle Model ..... 32
2.8 Projecting Forces onto the Needle ..... 32
2.9 Integrating Needle Forces ..... 33
2.10 Discussion ..... 34
3 Mesh Generation ..... 35
3.1 Prostate Surface Construction ..... 35
3.1.1 Treatment Planning ..... 36
3.1.2 Contour Extraction via DICOM-RT ..... 36
3.1.3 Culling the Contour Nodes ..... 36
3.1.4 Obtaining the Surface Definition ..... 37
3.2 Meshing the Region ..... 39
3.3 Results ..... 40
3.4 Local Element Numbering ..... 42
3.5 Mesh for the Needle Model ..... 44
3.6 Discussion ..... 46
4 Ultrasound Synthesis ..... 47
4.1 Patient Data Collection ..... 47
4.2 The Shift and the Torque ..... 48
4.3 Problem Definition ..... 51
4.4 A Direct Approach ..... 52
4.5 Inverting the Problem ..... 54
4.6 Image Interpolation Results ..... 56
4.7 Needle Overlay ..... 57
4.8 Discussion ..... 60
5 Implementation ..... 62
5.1 Program Overview ..... 62
5.2 Simulation Structure ..... 63
5.3 Tissue Model Initialization ..... 63
5.4 Needle Model Initialization ..... 65
5.5 An Iteration ..... 66
5.6 Notes ..... 69
5.7 Discussion ..... 70
6 Conclusion ..... 71
6.1 Summary and Contributions ..... 71
6.2 Future Work ..... 72
Bibliography ..... 73
A Finite Element Method ..... 79
A. 1 Definitions of Stress and Strain ..... 79
A. 2 Deformation ..... 81
A. 3 Linear Elastic Behaviour ..... 82
A. 4 Discretization ..... 83
A. 5 Element energy ..... 86
B Boundary Condition Change ..... 89

## List of Tables

3.1 Mesh quality. ..... 43
5.1 Tissue FEA variables. ..... 65
5.2 Needle FEA variables. ..... 66

## List of Figures

1.1 The anatomy of the prostate region. ..... 1
1.2 Prostate brachytherapy operation. ..... 3
1.3 Radioactive seeds implanted during brachytherapy. ..... 3
1.4 A representation of the 3D volume and two TRUS B-scans ..... 4
1.5 Fluoroscopy images from a brachytherapy procedure. ..... 5
1.6 The brachytherapy needle. ..... 6
2.1 Automaton describing the stick-slip friction model of a contact node. ..... 20
2.2 Tissue and needle-base axes in our implementation. ..... 20
2.3 An instance of contact node local frames lying on a bent needle. ..... 21
2.4 Subregion remeshing for a new element penetration. ..... 28
2.5 Four elements neighbouring to node $P$ which is to be repositioned. ..... 29
2.6 Projecting the working node forces onto the needle knots. ..... 33
2.7 Integration of knot forces and torques on the base. ..... 34
3.1 Node culling example. ..... 37
3.2 Node culling with a threshold of 4 mm on a sample data. ..... 38
3.3 Nuages surface output. ..... 39
3.4 Mesh visualization ..... 41
3.5 Mesh quality analysis. ..... 42
3.6 A right-handed tetrahedron ..... 43
3.7 Brachytherapy needle deflection under a lateral tip force. ..... 44
3.8 Needle mesh topology. ..... 44
3.9 Mesh topology of a section for two different designs. ..... 45
3.10 Lateral tip deflection with respect to number of sections. ..... 45
4.1 CT images after a brachytherapy procedure ..... 48
4.2 MRI after a brachytherapy procedure. ..... 48
4.3 Prostate shifting into the bladder during a needle insertion. ..... 49
4.4 The prostate region and the TRUS images in 2D. ..... 50
4.5 Demonstration of the structural deformation during needle insertion ..... 50
4.6 A sketch representing the 3D deformation of the initial TRUS images. ..... 51
4.7 Demonstration of the structural deformation during needle insertion. ..... 52
4.8 Determining whether a point is inside a tetrahedron. ..... 53
4.9 Ultrasound imaging plane transformed into the nominal mesh. ..... 55
4.10 Ultrasound plane with deformed tetrahedra. ..... 56
4.11 Interpolation results and needle overlay. ..... 58
4.12 Comet tail and multiple reflections artifacts of the needle. ..... 59
4.13 Needle shaft. ..... 60
5.1 Matlab interface running the simulation. ..... 64
5.2 A main iteration flowchart of needle interacting with the tissue. ..... 66
5.3 The simulation forces and timings. ..... 69
A. 1 Resultant force and moment vectors. ..... 79
A. 2 Components of stress on an infinitesimal cube. ..... 80
A. 3 A 2-D discretization example of a body $\Omega$ into finite elements $\Omega^{i}$. ..... 84
A. 4 A tetrahedral element and its corners in local numbering. ..... 84

## Glossary

| $(\phi, \theta, \psi)$ | Roll-pitch-yaw angles |
| :--- | :--- |
| 2D | Two-dimension(al) |
| 3D | Three-dimension(al) |
| BCC | Body Centered Cubic |
| BCCA | British Columbia Cancer Agency |
| BEM | Boundary Element Method |
| bNED | biochemical no evidence of disease |
| CDT | Constrained Delaunay Tetrahedralization |
| CG | Conjugate Gradients |
| CT | Computed Tomography |
| DICOM-RT | Digital Imaging and Communications in Medicine - Radiation Therapy |
| DOF | Degree of Freedom |
| DW | Distance Weighted |
| FEA | Finite Element Analysis |
| FEM | Finite Element Method |
| HDR | High-dose-rate (brachytherapy) |
| MRI | Magnetic Resonance Imaging |
| NURBS | Nonuniform Rational B-Splines |
| PNN | Pixel Nearest Neighbour |
| PSA | Prostate Specific Antigen |


| PTV | Planning target volume |
| :--- | :--- |
| RBF | Radial Basis Functions |
| TRUS | Transrectal Ultrasound |
| US | Ultrasound |
| VNN | Voxel Nearest Neighbour |
| VR | Virtual-Reality |
| $\underline{\boldsymbol{x}}$ | Mixed force-displacement outputs of working subsystem |
| $\underline{\boldsymbol{y}}$ | Mixed force-displacement inputs of working subsystem |
| $\boldsymbol{f}$ | Force vector |
| $\boldsymbol{K}^{-1}$ | Inverse global stiffness matrix |
| $\boldsymbol{K}_{H}$ | Condensed system matrix |
| $\boldsymbol{K}_{W}$ | Working matrix |
| $\boldsymbol{K}_{W}^{\prime}$ | Updated working matrix |
| $\boldsymbol{u}$ | Displacement vector <br> $m$ |
| $n$ | The number of neighbouring nodes of a node to be relocated |
| $w$ | The total number of nodes in a tissue mesh |
| $\boldsymbol{m}$ | Number of working nodes |

## Acknowledgements

- First of all, I would like to thank my supervisors Tim Salcudean, Robert Rohling, and Simon DiMaio for their guidance. In this journey of two years, their support has always been a positive motivation for me.
- I would also like to thank all my lab mates: Daniela, for her always being kind and helpful in whatever the topic of my problem is; Danny, for making our hospital visits enjoyable; Julian, for his endless "literal" help; Giselle, for making the lab environment fun; to Emre, for our valuable discussions; Ehsan, for always sharing his vast algebraic knowledge with me; and to all others including but not only Nicu, Hani, Xu, Sara, Hassan, Reza, ...
- I am also very grateful to Dr. Jim Morris and Dr. Mira Keyes. Their expertise in the clinical setting was invaluable and necessary in order to realize this project. And, without the help of the friendly staff of the BC Cancer Agency, the data collection would not be easy at all.
- I appreciate the logistic and psychological support of my fellow Johannians through the process of (especially writing of) this thesis.
- I definitely owe my being here to my parents whom I should acknowledge the most. Their endless support has always been with me.


## Chapter 1

## Introduction

### 1.1 Prostate Cancer

The prostate (see Figure 1.1) is a walnut-sized gland. It lies below the urinary bladder and surrounds the urethra. It adds important nutrients to the semen, which is the fluid containing sperms. The rectum wall lies just a few millimeters behind the gland and lying on each side of their interface are nerves and blood vessels. The urinary sphincter is a muscle just beyond the prostate and it functions to prevent involuntary leakage of urine. Finally, the vas deferens, a tube originating from each testicle, conducts the sperm to the urethra inside the prostate.


Figure 1.1: The anatomy of the prostate region.
Prostate cancer is the most common cancer among men in Canada, with 20,100 new cases and more than 4,200 deaths estimated in 2004 [1]. Despite the high rate of the disease (it will afflict 1 in 8 Canadian men in their lifetime), it is a relatively less aggressive and slow processing cancer type especially in the elderly. That is why the decision of treatment method has to be
made carefully considering involved risks and life expectancy, and in many cases an expectant management ("watchful waiting") may be opted for.

Some treatment options are: prostatectomy, the surgical removal of the prostate; chemotherapy, the use of drugs; cryotherapy, a new technique of freezing the gland in situ; hormone therapy, which downsizes the prostate and suppresses the cancer; and radiation therapy. First type of radiation therapy is the external beam radiation which focuses beams on the region. The second type, called brachytherapy, applies radiation internally. Both radiation and cryo approaches kill cancerous cells by basically destroying the prostate glands which are later replaced by scar tissue. Usually combined treatment modalities are employed rather than a monotherapy in order to eliminate all cancerous tissue within and around the prostate also eradicating extracapsular disease and seminal vesicle involvement. An important tool for assessing the level of the disease is the serum prostate specific antigen (PSA) which can be tested in a blood sample.

### 1.2 Brachytherapy

Brachy is a Greek term for short distance. High-dose-rate (HDR) brachytherapy refers to the temporary application of radiation intra-operatively using a powerful source in an inserted needle. For early stage locally confined prostate cancer, low-dose-rate brachytherapy which is the permanent implantation of low energy radioactive pellets (seeds) is often the treatment of choice. Its main advantage over other methods is that the delivered dose is very conformal to the prostate and it spares normal tissue. Furthermore, this cost-effective solution maintains urinary continence and it has a short convalescence time. Long term results from the pioneering Seattle group shows the effectiveness of this treatment. They reported a promising 10 year biochemical no evidence of disease (bNED) at 84-93\% for patients with no more than one favourable risk factor $\left(\right.$ PSA $>10 \mathrm{ng} / \mathrm{mL}$ or Gleason ${ }^{1}>6$ ) [2].

In spite of the various treatment options, permanent seed implantation (from now on referred as brachytherapy) combined with other modalities has become common because of the flexibility in individual customization of source placement and target volume. A right choice of plan with proper application is very likely to eradicate the cancer while preserving the healthy tissue. However, despite the low risk of brachytherapy, seed placement errors, which may lead to subsequent complications such as urinary incontinence and impotence, are still common. In [3], an experienced physician implanting seeds in 20 patients achieved an average placement error of 0.63 cm , a substantial error of $21 \%$ of average prostate diameter ( 3 cm ). In addition, a suboptimal application resulting in an undesired dose distribution at the target volume may in turn decrease the effectiveness of the treatment. Therefore, while not as invasive as surgery, brachytherapy requires significant skill on the part of the performing physician.

The implant procedure has numerous preoperative and postoperative components. Most patients are managed preoperatively with external beam radiation to treat locally disseminated

[^0]disease and/or with neoadjuvant hormonal therapy to downsize the gland and inhibit the spread. For planning the individual seed arrangement a preimplant ultrasound (US) volume study is done several weeks before the procedure. Both the volume study and the procedure are performed with the patient in the dorsal lithotomy position (lying on his back). A sketch of the procedure setup is shown in Figure 1.2(c). A transrectal ultrasound (TRUS) probe which is positioned using a stepper unit mounted on a stabilizer captures images at 5 mm increments (see Figure 1.2(b)). The template grid (which is described below) attached to the stepper also discretizes this volume transversally at 5 mm intervals.


Figure 1.2: Prostate brachytherapy operation.

For planning, the ultrasound prostate volume is expanded with margins to a planning target volume (PTV), which encompasses likely extracapsular disease. The decision of PTV seed arrangement is made by a radiation oncologist using special software to evaluate parameters, e.g., dose distribution. Prior to the procedure, the sterilized seeds (see "standard seeds" in Figure 1.3) are loaded into needles. The BC Cancer Agency (BCCA) uses OncoSeed ${ }^{T M}$ that is a welded titanium capsule containing radioactive iodine isotope ${ }^{125} \mathrm{I}$ (with a half-life of 60 days) absorbed onto a silver rod.


Figure 1.3: Radioactive seeds implanted during brachytherapy (reprinted from [4]).

Fluoroscopy is a medical imaging modality, which is also utilized in brachytherapy, to view the anatomy by means of X-ray. The operation room with the a C-arm (for fluoroscopy) and

US setup can be seen in Figure 1.2(a). Although the C-arm can rotate for different imaging directions, fluoro-images are only taken along the posterior-anterior axis in brachytherapy. The intra-operative TRUS probe is set up in the same way as in the preimplant volume study so that the prostate can be registered to the implant plan in the beginning of the procedure. Loaded needles are inserted via a template grid (see Figure 1.2(c)) the coordinates of which are given by the preplan. Needle positioning is confirmed using the TRUS (see Figure 1.4) and fluoroscopy (see Figure 1.5). For guidance, white dots denoting template grid holes are overlayed onto the ultrasound images by the TRUS hardware as in Figure 1.4(b). Since there is a risk of Xray overdose, the number of fluoro-images taken during a procedure must be kept low (20-50). Consequently, TRUS becomes the only modality supplying continuous feedback to the physician.

Before the procedure the patient is anaesthetized. If anaesthesia is not safe, the spinal blockade is applied so that the patient is temporarily paralyzed below waist. First the physician fixes the stabilizer and the precision stepper which are used to position the ultrasound probe. The stepper is calibrated by matching the TRUS images with the ones that were taken in advance. These manual adjustments are done by visually comparing the printed images with the intraoperative TRUS imaging. It takes relatively long time and it is subject to the experience of the physician. Then the fluoroscope is quickly adjusted and the radioactive seeds are taken into the operation room.

The brachytherapy needles are loaded with up to six seeds each prior to the procedure. If all seeds in a needle are spaced by 1 cm , then a special type of seed packaging (the coated ones at the bottom in Figure 1.3) can be used. The BCCA uses the brand RAPID Strand ${ }^{\mathrm{TM}}$ - Rigid Absorbable Permanent Implant Device). This manufacturer wrapped strands are preferable since they are less likely to jam in the needle. They also stay at the implanted location and orientation whereas the individual seeds may rotate during retraction or can travel through blood vessels. Custom spacing is achieved by adding spacer between individual seeds. The needles are inserted to their targets through a grid template. The grid hole and insertion depth are double-checked by a nurse who is reading the therapy plan to the doctor. The stepper may need to be readjusted during the procedure because of the prostate swelling. The doctor may also modify the procedure


Figure 1.4: A representation of the 3D volume with TRUS layers and a needle (a), and two B-scans from a procedure.


Figure 1.5: Fluoroscopy images from a brachytherapy procedure.
plan slightly and/or use extra seeds at the end of the operation to correct and improve the dose distribution.

During a procedure $\approx 100$ seeds are implanted using 20 to 25 needles. The needles are 20 cm in length and quite flexible with a significant tip bevel. A needle consists of an outer cannula (that holds the seeds) with black stripes marking every centimeter and a stylet that tightly fits in this cannula (see Figure 1.6). So, the seeds can be released by pulling the needle cannula while holding the stylet steady. Just before that, measuring the distance from the cannula base to the template and to the stylet holder ensures the depth of the current implant and the queueing of the first seed to the needle tip, respectively.

An arrow on the cannula base marks the direction of tip bevel. Using this bevel and the needle flexibility, the physician can steer the needle. This can be done to refine the grid discretization by reaching in-between template holes, if the physician wishes to make slight modifications to the preplan. Moreover, a direct insertion cannot reach behind the pubic arch where steering becomes necessary.

Proper steering places seeds away from certain anatomical landmarks, e.g. the urethra and the rectum, while not overdosing or underdosing a region. However, this requires the experience of many procedures. Successful implants rely upon not only a thorough comprehension of the relative 3D seed locations and surrounding anatomical structures, but also taking their deformation during needle insertion into account. These challenges reveal the need and importance of brachytherapy training.

### 1.3 Need for Training

The lack of direct visual feedback makes the comprehension of the needle path with respect to the anatomical layout rely only upon the medical imaging modalities, ultrasound and fluoroscopy, and the kinesthetic feedback on the needle base. The fluoroscopy gives a clear view of seeds


Figure 1.6: The brachytherapy needle.
and the TRUS imaging supplies continuous information about the region. In addition to those imaging modalities, haptic ${ }^{2}$ feedback also gives important information to the physician about which medium the tip is in, when the prostate membrane is punctured, or whether the pubic arch is hit. However, correctly processing the images, the kinesthetic feedback, and their combined information in terms of the anatomy of interest requires training.

As mentioned before, there is a limit on the number of fluoroscopy images that can be taken during brachytherapy because of the X-ray radiation. Aside from that limitation, fluoroscopy does not convey all 3D geometry (e.g., the height info) either. On the other hand, unfortunately, the prostate, the pathology, the seeds, and the needle are not always clearly visible in TRUS images. Furthermore, the procedure must be completed in a relatively short period of time to minimize edema and anesthesia side-effects, while the best possible seed distribution may require intra-operative plan modification which also adds up to the complexity.

In order to develop all those skills, medical residents use mannequins, animals, or cadavers, which are either not very realistic or may involve ethical issues. The training of a resident also involves observation and practice on real patients, which may lead to irreversible consequences even though procedures are supervised by an experienced physician.

The identification of the need for better training schemes led us to consider alternative noninvasive solutions. Based on this motivation, this thesis proposes a simulator of the brachytherapy procedure, on which the medical residents can train and the expert physicians can excel their skills while experimenting new techniques.

As outlined above, a brachytherapy simulator should convey the haptic and imaging feedback to the user. Thus, a tissue model interacting with a needle, which is manipulated by the user holding on its base, is needed so that the reaction forces of this model can be applied back onto the user's hand via the base. The imaging feedback may be ultrasound and/or fluoroscopy.

[^1]Due to its more frequent referral by the physician during the procedure, ultrasound has been incorporated first in our simulator. Its continuous representation of tissue deformation was also coherent with our goal of tissue deformation modeling. For realistic TRUS feedback, images need to be generated according to the current placement of the tissue and the needle. This required a model for the deformed tissue image generation and needle representation on it.

### 1.4 Related Work

Just as a flight simulator is used for training pilots and it helps in identifying the complications of the current and proposed procedures, a medical simulator is built to recreate the realm of a surgery or examination to be used in training and testing. In this area there exists significant amount of previous work including epidural lumbar puncture [5], breast biopsy [6], interstitial brachytherapy [7], prostate cryotherapy [8], prostate brachytherapy [9], hepatic surgery [10]. There are also current commercialized products on laparoscopy [11, 12, 13], endoscopy [11, 12], and intra-vascular access [11, 12].

Our proposed needle insertion simulator for prostate brachytherapy includes a tissue deformation model. Generating a tissue mesh for this model and the interaction of the model with a brachytherapy needle are two issues that are addressed in this thesis. Finally, ultrasound synthesis as a visual feedback is discussed. Some related studies in those areas are mentioned throughout the rest of this chapter.

The realism of a medical simulator is drastically affected by the tissue model of interest. Although the interconnection of muscle fibers, veins, nerves, etc. makes the prediction and generalization of tissue behaviour almost impossible, still some major characteristics (e.g., deformation) can be approximated for certain cases with available techniques $[9,10,14,15,16]$. For the simulation of these characteristics first a tissue model of the region of interest is needed. Section 1.4.1 presents some of them.

Earlier attempts for haptics in medicine were simple approaches such as look-up tables [17]. In the course of time technological advances have enabled the use of haptics for physically-based interaction in medical simulations with complex control loops requiring fast update rates. Haptic implementations including palpation, incision, and needle insertion require a stable and realistic contact handling which has been a challenge in surgical tool simulations. Hirota et al. [18] recently proposed a contact method for finite element analysis (FEA) of non-linearly elastic objects and applied it to a knee bending simulation. Friction models for tissue interaction is another concern of medical simulation. Section 1.4.2 mentions methods that were applied in needle-tissue interaction.

Owing to vision being our primary sense, most medical simulators have focused on graphics. Many created a virtual-reality (VR) environment that renders the region of interest from the surgeon's eyes [10]. Fuchs et al. [19] overlayed oriented ultrasound images onto the patient using VR goggles during US examinations. Some researchers preferred rendering the 3D anatomy
including the obscured internal structures in their simulations [20]. The implementation and realism of rendering methods were largely dealt in computer graphics. However, during a needle insertion nothing more than the tissue surface can be seen. This makes it completely dependent on medical imaging techniques such as ultrasound.

Hahn et al. [8] developed a graphical interface for cryotherapy without haptics. They cut the desired US slices from a 3D volume data acquired by a 3D US device and overlayed the freezing probes as white dots. Kimura et al. [20] designed a brachytherapy simulation with US, fluoroscopy, and a 3D computer graphics rendering of the region. They added the needle as a black line on fluoro image and as a white disk on US. They also supplied a simple haptic feedback, a peak for puncture. Unlike the above mentioned ones, Wang and Fenster [21] proposed a modelbased approach to brachytherapy simulation. For a 3D graphical rendering of the region they used a modified version of the 3D ChainMail method described in the next section. They loosely coupled this deformation model with haptic feedback of only the prostate.

Some recent research on improving brachytherapy procedures are by Schneider et al. [22] and Wei et al. [23]. The former proposes a new technique for inserting the needles through the rectum via a partial sheath surrounding the TRUS probe in order to maximize the seed positioning and the comfort to the patient. The latter introduces a robot arm guiding the orientation of the template in order allow oblique insertions. A major part of their work focused on finding the needle on US volume data gathered by a rotational fan scan using a side-firing transrectal probe.

### 1.4.1 Tissue Modeling

The trend toward computer-based medical simulators brought a high demand for modeling the complex characteristics of the tissue. Neglecting the inhomogeneities, anisotropy, relaxation, and hysteresis, soft tissue is generally considered to be a non-linearly deformable elastic material. In computer graphics, deformation modeling has been studied for noisy data fitting, clothing, facial expressions, and human/animal characters. [24] gives a survey of deformable modeling in computer graphics.

In tissue modeling, one of the proposed heuristic approaches is the ChainMail algorithm introduced by Gibson [25]. It estimates the object as small patches which only transfer their motion to their neighbour when their displacement is higher than a preset value. ChainMail does not supply a model for haptic force feedback calculations and the deformations are not realistic.

A physically-based approach, the mass-spring model, approximates the body as a set of masses connected via springs (and possibly dampers). Waters [26] was the first to model the muscles of the face and generate gestures using this model. Later, Terzopoulos and Waters constructed a 3-layer dynamic mass-spring mesh for face in [14]. Although some degree of realistic visual and haptic feedback can be achieved as shown in cloths made of double cross springs by Promayon [27] and laparoscopic simulator by Downes [15], the performance of massspring models highly depend on their spring network structure. Even after proper adjustment of the parameters, the models can only accommodate certain interactions and the instability is
a major problem.
Unlike the above mentioned approaches, the finite element method (FEM) and boundary element method (BEM) are based on continuum mechanics. James and Pai introduced BEM in [28] by solving the boundary integration of Navier's equation formulating the linear elastostatic objects. However, this method can only accommodate one material in a model and only solves for the boundary deformations/forces. In their later work [29], the same authors achieved realtime deformable textures using graphics hardware. They stored precomputed modal vibration models in graphics card memory and in real-time excited them according to rigid body motions.

Continuum mechanics defines the deformation of an elastic body such that the equilibrium of the internal strain energy and the work done by external forces is preserved. Although a spatially continuous model is possible, because of the computational limitations in solving them on computers the implemented forms are ultimately discrete. Thus, the object is subdivided into smaller pieces and numerical interpolation/integration techniques are employed to approximate the system at discrete nodes. This is called the finite element method. It makes the solution performance dependent on this discretization (mesh). Meshing techniques are summarized in Section 1.4.3 and our approach is given in Chapter 3.

The earlier work of Terzopoulos et al. [30] showed the potential of FEM and finite difference method, which a special form of FEM. Cotin et al. [31] demonstrated a real-time visual performance in an FEM-based surgery simulator using linearly elastic materials with quasi-static deformations. Zhuang and Canny [32] reached real-time rates for large meshes using quadratic strain and dynamic FEM, which incorporates mass and damping effects onto the basic stiffness equation. They used a lumped mass matrix and exploited the small stiffness of the soft material to use explicit Newmark integration scheme for large time steps ([33]). Picinbono et al. [10] introduced an incompressibility constraint on a nonlinear dynamic mass-tensor model with lumped mass and damping and they simulated anisotropy.

Both above mentioned models by Zhuang and Picinbono et al. use St. Venant-Kirchoff elasticity which is the non-linear (non-simplified) version of the linear elasticity described in Appendix A. Debunne et al. [34] implemented a space-time adaptive mesh refinement on top of this same non-linear dynamic simulation scheme.

Wu et al. [35] and Nienhuys [36] attempted other elasticity formulations (e.g., neo-Hookean). [35] implemented an adaptive mesh refinement using offline progressively coarsened meshes. [36] also compared some iterative solution techniques such as conjugate gradients (CG) and applied some 2D mesh modification and 3D mesh refinement techniques on incision and needle insertion examples.

Conceptual hypotheses can be easier experimented in predefined media (phantoms) than complex tissue structures. The concepts laid out and validated through these phantom studies are then considered to be applicable to various tissues using their biomechanical properties reported in literature. In a recent study, Kerdok et al. [37] validated their finite element deformation
using the volumetric displacement data ${ }^{3}$ of markers embedded in a rubber cube and tracked by CT (Computed Tomography).

### 1.4.2 Needle Insertion

The simulation problems related to needle insertion slightly differs from the other tool interactions in a couple of ways: it does not manipulate only the organ surface; the friction is an effective force during its interaction; and it is flexible unlike mostly-rigid surgical tools. The pioneering work of DiMaio and Salcudean [38], solved the above mentioned issues in 2D using a condensation technique, stick-slip friction with low-rank boundary condition switch, and local frame low-rank matrix update, respectively. The related work on those issues are presented below.

In conforming FEM the needle surface forms a boundary condition for the tissue. Although Berkley et al. [39] showed a virtual coupling is possible by interpolating the forces/displacements, this interpolation brings an overhead and also the boundary criteria to be applied become vague. Moreover, the increase in the number of manipulated nodes negatively affect the speed of condensation techniques mentioned in Section 2.1.2. Therefore, a conforming tissue model with nodes falling on the needle shaft was chosen.

To retain the tissue mesh nodes on the needle three approaches were implemented: Alterovitz et al. [9] and Nienhuys [36] insert new nodes while subdividing elements; Alterovitz et al. [9] and DiMaio [40] relocate nodes onto the needle path; and the haptic implementation of [38] forces mesh nodes to be on the needle which locally deforms the tissue. The tissue nodes on the needle can efficiently be solved using the condensation technique introduced by Bro-Nielsen and Cotin [41]. In a recent work, Gosline et al. [42] introduce fluid pockets onto DiMaio's simulation [40] by adding their boundary force constraints onto this condensed working matrix.

Generally, the online mesh modification is a major problem of FEM also required during incision [36]. Nonetheless, more direct approaches such as simply removing elements still exist. For instance, Cotin et al. [43] implemented this on their hybrid elastic model with partly quasistatic and partly dynamic elements.

Needle insertion forces were studied by many researchers. Some earlier methods employed look-up tables such as the 1-DOF (degree of freedom) epidural block simulator of Hiemenz et al. [17]. Simone and Okamura [44] used a modified Karnopp model for needle friction estimation during in vitro insertions into the liver.
[45] attempted a cantilever-beam-like needle flexibility to model the bevel deflection from in vitro muscle insertions, however in their work this model did not perform well. It makes an assumption of a distributed lateral force along the shaft dependent only on the insertion depth and the needle diameter as opposed to DiMaio's proposal of one deflection force at the tip as a function of velocity in [40]. O'Leary et al. [46] also attempted an insertion force modeling of bending and friction, however their approach of using the combined force on the base did not

[^2]lead to significant results.
Although Okamura et al. [47] modeled the in-vivo tip cutting force and the insertion-retraction hysteresis effect, this and many other studies only considered the needle base forces. DiMaio and Salcudean [16] decoupled the shaft friction from tip cutting force and validated velocity dependent models physically. They were the first to model a velocity dependent shaft friction and tip cutting force through FEA of phantoms during needle insertions. In their later work [48], they present a needle steering scheme using potential fields. If needle steering is divided in two, steering using tip bevel and redirecting the tip by base manipulation, they explored the latter one. For needle flexibility, they employed a large-strain linear elastic model, that uses the GreenLagrange strain tensor.

Webster et al. [49] recently showed that the highly flexible needles follow a path of constantcurvature for a fixed bevel rotation. They used a 3-DOF nonholonomic bicycle model for the tip bevel and found its parameters experimentally via insertions into semi-transparent and stiff phantoms. Alterovitz et al. [50] incorporated a simplified version of this concept into their needle insertion simulator.

### 1.4.3 Meshing Techniques

Tetrahedral meshes are commonly used in many areas including solid modeling [51], haptics [34], fracture [52], fluid flow [53], heat transfer [54], surgical simulations [55], and biological tissue modeling such as the brain [56], the knee [18], the liver [35], and the uterus [57]. Therefore, generation of meshes with desirable characteristics has always been an interesting research topic. However, in this broad field mesh requirements highly depend on the application. The optimum mesh for soft tissue modeling with large deformation, solid modeling with small deformation, and fluid flow and heat transfer modeling with no deformation differ significantly. Likewise, the quality measures for mesh/element evaluation also depend on the problem type [58]. Many solutions were proposed for specific areas, which would not be considered even a suboptimal solution in another field. For example, it is desirable to have anisotropically compressed elements in boundary layers of a mesh for an Eulerian fluid flow simulation [59], since the solution gradient is obviously smaller in the direction of the flow. However, those elongated elements would result in anisotropy in a deformation simulation. Therefore, we first defined the desired characteristics of a deformable soft tissue mesh for prostate brachytherapy in Chapter 3.

Although there are various algorithms for tetrahedral mesh generation, most of them are based on and classified in one of the following three methods. The algorithmically simplest one is the octree technique, which was originally developed at the Scientific Computation Research Center of Rensselaer Polytechnic Institute [60]. It is the recursive division of the model space into smaller elements until the desired resolution is reached for the elements that are cut by the object boundary. Another meshing technique is the advancing fronts [61] which progressively builds the tetrahedra starting from the triangulated surface and maintaining an active front that moves inwards by the formation of new elements. In 3D, for each triangular facet on
the moving front a fourth node position for a well-shaped tetrahedron is calculated and the corresponding element is added. Before this node addition, the current nodes are also considered for a well-shaped element formation. This technique initiated with a poor surface triangulation will engender ill-conditioned tetrahedra. Moreover, its performance degrades when the fronts are merging which may as well occur near the surface with high curvatures. The popular FEA software ANSYS ${ }^{\circledR}$ includes a version of Lo's advancing fronts implementation [62].

The third and the most exploited tetrahedral meshing technique is based on the criterion set by Delaunay [63] in 1934. Until the work [64] in 1981, this well-known criterion was not utilized for mesh generation. The Delaunay tessellation of a set of vertices $V$ in 3D has the property that the circumsphere of any tetrahedron encloses no other vertex of $V$. Since this criterion is just a connectivity definition, not a meshing algorithm itself, extra tetrahedralization methods are required for mesh generation. These methods basically differ by their decision mechanisms on the position of the new point that is to be introduced. For example, a simple approach can be filling the space with a regular grid. More complicated approaches usually first tessellate the triangulated surface nodes and then considering the quality and relative positions of the current elements, they insert nodes to divide the ill-conditioned tetrahedra. Shewchuk [65] introduced a guaranteed quality triangulation based on the node insertion at the element circumcircle centers and he implemented this in a 2D mesher ${ }^{4}$. The method [67] developed at INRIA employs a incremental node insertion from a list of pregenerated candidate vertices until a background sizing function is satisfied. This was also implemented in their meshing software.

Mesh postprocessing schemes such as smoothing, cleanup, and refinement can be applied to improve element qualities. Smoothing refers to the slight relocation of the mesh nodes. Some of the smoothing techniques use averaging, optimization-based, or physically-based (e.g. spring interaction between neighbours) methods. Cleanup is the step where element connectivity is optimized using edge/element swaps (flips). It is used either for shape improvement, which leads to better-quality elements, or for topological improvement, which optimizes the number of edges sharing a single node (node valence or degree). Refinement is the reduction of the local element sizes by node insertion for the purpose of capturing a local physical phenomenon or simply increasing the mesh quality.

One drawback of the postprocessing may be that most of the methods above violate the Delaunay criterion of a mesh. Successful adaptive approaches were proposed based on the combination of mesh postprocessing steps and an iterative application of the Delaunay tessellation iteratively. Nevertheless, Delaunay techniques do not perform well in 3D, since many straightforward 2D methods such as edge flips become excessively complicated when adapted to 3D and may be intractable due to the non-polynomial complexities of the algorithms involved. In 3D, iterative edge flips may even not terminate. Moreover, slivers (very flat elements with negligible volume) become another source of problem in 3D despite the sliver removal steps. [65] provides an overview of these methods including a discussion on why theoretical results often fail

[^3]in practice.
It is very common that objects to be meshed are not convex and they contain holes or subregions (e.g., different materials) separated by internal boundaries. This problem is referred as boundary mesh generation or boundary conformity. One solution is a constrained Delaunay tetrahedralization (CDT), which can generate a mesh that has no facets cutting the internal boundary. A general approach is the facet recovery by edge swaps. However, it is extremely complicated and there are even cases where no valid tessellation is guaranteed without new point insertion [68]. During the Delaunay criterion check of a CDT, circumspheres passing to the other side of a constraining boundary are ignored. Therefore, elements of CDTs are not necessarily Delaunay.

Nonuniform Rational B-Splines (NURBS) is a powerful way of geometrical representation. In meshing, NURBS can represent the boundaries of a region as a set of parametrical curves instead of line segments between nodes. Since the prostate boundary has low curvature and our segmented nodes are already finely spaced, we did not employ any NURBS techniques.

A recent study [57] achieves an almost uniform tetrahedralization as follows: first the boundary cutting elements on an initial uniform mesh are subdivided; then the tetrahedra located outside are removed; and finally the positions of the remaining vertices are iteratively optimized considering the element quality measures. However, Chapter 3 mentions the FEA drawbacks of uniform meshing.

Detailed surveys of mesh generation can be found in [69] and online at [70]. In general, the initial surface mesh is preserved and the number of elements is comparatively large. The high demand by the recent soft tissue studies lead to a lot of research in the area. There are methods promising well-conditioned meshes for subsequent large deformations, e.g., in [71] the refinement of an initial mesh with body-centered-cubic (BCC) crystalline structure towards the surface. However, in those mainly the surface representation and interaction are focused. Therefore, these coarse inner structure is likely to fail during a haptic interaction with inner nodes. On the other hand, there is also related prior work (e.g., Debunne et al. [34]) generating good-quality elements locally from coarse meshes using adaptive meshing techniques.

The types of elements are named according to the interpolating basis functions used. For instance, in our simulation TET4 elements are used. This name refers to a tetrahedron where inner displacements are approximated linearly using its four corners. Higher order shape approximations (e.g., TET10 with 6 more nodes on the edges) obviously perform better with a computational complexity drawback.

### 1.4.4 Ultrasound Imaging and 3D Reconstruction

Ultrasound offers tomographic imaging capability in a rapid (10-100 images/sec), noninvasive manner [72]. Tremendous progress over the past few years has improved its image quality and visualization clarity significantly. During brachytherapy the only continuous imaging feedback is the TRUS. It gives information on the positions of the needle tip and the already implanted seeds
with respect to the volume. That is why this imaging modality is crucial in decision making and so it should be incorporated in our simulator.

The problem to be tackled is the generation of TRUS images in a deformed volume, the initial undeformed state of which is known in terms of regularly sampled B-mode TRUS data. If we were to divide our problem into two parts, volume deformation and the synthesis of TRUS images out of this deformed volume, the second part becomes a slicing problem of 3D (reconstructed) ultrasound data. The first part is attempted using FEM-based techniques (basis functions) in Chapter 4. In order to gain more insight into the second part, let us present a summary of prior work on 3D ultrasound volume reconstruction.

Traditional single 2D B-mode ultrasound mostly relies on the operator's interactive ability of acquiring a properly oriented image of the target structure for diagnosis. This highly depends on 3D comprehension of the anatomy and its 2D ultrasound representation. On the other hand, 3D ultrasound can be sliced in any direction, can be rendered for easier visualization, or can be stored and reviewed at a later time. In this concept, reconstruction refers to the generation of 3D volume data on computer using the 2D slices acquired from the region of interest. So, basically this 3D data can then be displayed using computer graphics techniques or resliced in order to generate 2D B-mode images. Nevertheless, note that in our system this 3D data changes dynamically as the tissue deforms and this necessitates additional steps before reslicing if the same approach is followed.

For volume reconstruction, scanning can be done in various ways. [72] classifies the major ones as:

Mechanical: The transducer array or the entire probe is moved mechanically. Regular sampling achieves an accurate geometry, however, the bulky mechanisms are generally not practical.
Position sensing: The probe position is tracked using sensors (e.g., magnetic, infrared). The probe is easier to manipulate, but the data is harder to reconstruct due to sensing errors.
Trajectory assumption: The operator moves the probe along a predetermined trajectory (e.g., spatially regular data sampling generated from slices sampled at regular temporal intervals during a constant velocity sweep). Distortions and inaccuracies are common in this method.

2D transducer arrays: A pyramid-like volume is scanned using dynamic beam steering. A relatively small region is scanned, but at a fast rate (suitable for echocardiography).

The reconstruction from 2D images can be done using feature-based or voxel-based approaches. The former segments the surfaces or the volumes in the acquired data, and only displays the simple anatomy. Although it is fast, subtle tissue information is lost and the segmented boundary can easily be erroneous. The latter puts each pixel at the corresponding 3D position. All information is retained, however, large data files must be manipulated and fast algorithms are needed.

Some of the factors that play a role in 3D reconstruction quality are the accuracy of the initial data position information and the selection of a coordinate system and a bounding box.

These problems, such as preprocessing of the data to minimize patient motion and breathing artifacts, have been tackled by several authors. However, since this thesis does not focus on the above mentioned issues, only the reconstruction phase (i.e., the resampling into a regular grid) is addressed here.

Reconstruction can briefly be divided into two stages, a bin-filling and a hole-filling, as suggested by Rohling et al. [73]. The former is also referred as compounding. It is the process of combining multiple pixel information for a voxel. This necessity arises either because of the voxel/pixel ratios greater than 1, or more commonly because of the overlapping B-scans. Detmer et al. [74], Rohling et al. [75], and Leotta and Matin [76] achieved some of the significant contributions in 3D freehand ultrasound compounding. Specifically, [75] introduces an imagebased registration approach of the compounded data, to avoid common problems (e.g., blurring and misregistered structures) in spatial compounding. Bin-filling is not applicable to our problem, since in our proposed initial data does not have more than one value at a coordinate. However, since the TRUS probe is moving through the volume many times during the whole procedure, effective methods that utilize compounding techniques are also applicable.

The hole-filling stage is the interpolation, where the values of voxels which have not yet been filled are to be determined. The interpolation can be done in several ways. Franke presents a review and comparison of scattered data interpolation methods in [77]. However, they are not very helpful in tracking features for our case of having a much greater distance between adjacent B-scans ( 5 mm ) than their internal resolution $(\approx 0.17 \mathrm{~mm}$ ). Another challenge is the complex characteristics of speckle on individual B-scans. The basics of speckle formation is given in the early work [78] of Thijssen and Oosterveld. Regarding the high dimensionality of the problem, simpler approaches are usually preferred. In addition to the common ones like voxel nearest neighbour (VNN) and pixel nearest neighbour (PNN), various others such as distance weighted (DW) [79], radial basis functions (RBF) [80], tetrahedral grids [81], and ellipsoid Gaussian convolution kernel [82] were also proposed. [73] provides a survey of wellknown reconstruction methods.

Visualization, one of the major challenges of 3D ultrasound, is approached by slice projection, surface fitting, segmentation, volume rendering and depth cue enhancement techniques. However, our problem is generating only a (2D) B-mode slice of the volume, therefore, visualization of 3D is not one of our concerns.

On the prostate side, 3D ultrasound shows some evidence of better prostate disease evaluation and more accurate volume estimation. It has the potential to improve the planning and evaluation of brachytherapy procedures. After its introduction by Watanabe [83] over 30 years ago, transrectal ultrasound had a very important role in diagnosis and intervention of various prostate diseases. Tong et al. [84] produced a 3D prostate volume rotating a side-firing transducer, although the modern prostate brachytherapy is performed using end-firing transducers.

One of the major difference of our problem from the above mentioned ones is the spacing between individual data slices. TRUS images have a 5 mm inter-slice spacing as opposed to an
only $\approx 0.17 \mathrm{~mm}$ pixel spacing on a slice. This prevents an efficient use of the typical interpolation approaches, since the features such as speckle change a lot in 5 mm and the speckle shape and distribution are intractable to model [78].

Although studies showed the potential of real-time visualization based on raw B-scans [85], the construction of a regular grid is still the preferred method. It enables further processing of the data with off-the-shelf algorithms. However our proposed method synthesizes the images directly from raw B-scans, since it takes much less time compared to the rest of the algorithm and the volume reconstruction method would bring huge memory requirements. Nevertheless, for a shorter computation time this method can also easily be applied on a volume which is reconstructed by interpolation.

Simple interpolation algorithms are successfully used on reconstructed volume sets to achieve B-mode slices. Aiger and Cohen-Or [86] implemented this idea in a commercialized application called UltraSim ${ }^{\text {TM }}$ ignoring the directional effects of ultrasound. [8] included a similar image generator as the ultrasound modality in their cryosurgery simulator. There are available commercialized 3D ultrasound systems doing such volume generation and slicing.

### 1.5 Proposed Simulator

A 3D needle-tissue interaction model is proposed for the specific application to prostate brachytherapy. In this interaction simulation, the coupling of a flexible needle with a linearly elastic tissue is achieved where the tissue is modeled as a soft deformable body by extending the quasistatic and linearly elastic FEA techniques presented in [40]. In this simulation implementation, the reaction forces on the base are computed while the needle base is being manipulated. A TRUS imaging method of the deformed tissue model is also proposed as a visual feedback modality.

### 1.6 Thesis Organization

Chapter 2 describes the FEA techniques used in simulation. The reader is referred to Appendix A and the cited books for a further insight on FEM. Then, Chapter 3 details the mesh generation methods attempted and the final mesh applied. In Chapter 4 the deformation in 3D ultrasound and its reslicing for intra-operative TRUS synthesis are discussed. The overall structure of the simulation with the data structures and implementation details are presented in the following chapter, Chapter 5. Finally, our achievements, conclusions, and possible future directions are laid out in Chapter 6.

## Chapter 2

## Needle-Tissue Interaction Model

Both the tissue and the needle are modeled using the finite element method. The reader is referred to Appendix A and related books [87, 88] for an introduction to finite element analysis (FEA). Many of the following approaches were extended to 3D from the work of DiMaio [40]. In brief, the tissue and the needle are modeled using FEM, the displacement of the needle is applied to a set of tissue model nodes, and the force feedback of this model is applied back to the needle FEM model. As the needle advances, the set of tissue model nodes that the forces are applied are updated when needle tip penetrates a new tissue element. Considering that these nodes slide along the needle their positions are updated using a stick-slip friction model. This model basically either fixes the node onto the needle shaft (by setting three displacement constraints) or lets it slide along (by setting two displacement constraints in lateral axes and a force constraint along the needle shaft axis). All of the above are discussed in detail throughout this chapter.

In here, our ultimate goal is to lay the grounds for a stable and realistic model of flexible needle and 3D soft tissue. To achieve this the following techniques are applied.

### 2.1 Stiffness Matrix Inversion and Condensation

For visualization of an FEM model, the displacements of some (e.g., the surface) or all of the model nodes are needed. In the proposed simulation, US is the visualization modality. So, the generation of the TRUS images requires the displacements of a subset of model nodes. Thus,

$$
\begin{equation*}
u=\boldsymbol{K}^{-1} \boldsymbol{f} \tag{2.1}
\end{equation*}
$$

needs to be solved, where $\boldsymbol{K}^{-1}$ is the inverse global stiffness matrix. The displacements are needed at 12 Hz considering the TRUS screen update rate during brachytherapy procedure. Although visualization is not an issue in haptic interaction and it can mostly be handled by a separate computer, in the proposed implementation, (2.1) is handled inside the haptic loop. This is the scheme followed by DiMaio in [40] and it was proven to work efficiently in his
implementation although for very large meshes speed issues may arise.
Various numerical methods can invert this $\boldsymbol{K}$ matrix relatively fast. For example, LU decomposition is a powerful tool considering the very sparse characteristic of the stiffness matrix. Reordering the rows/columns using Cholesky Decomposition or reverse Cuthill-McKee method can also improve the speed. However, for large 3D meshes, offline precomputation of this inversion is necessary for real-time performance. Even then, an online inversion might be needed if the mesh structure is modified such as an element is removed (e.g., incision) or a node is relocated (e.g., mesh adaptation in Section 2.6).

### 2.1.1 Inversion

Considering only a subset of displacements (e.g., the anatomical boundaries, seed locations, or elements on the TRUS plane) are needed, multiplying only the rows of $\boldsymbol{K}^{-1}$ corresponding to the these displacements would be enough. This would decrease the computations from $\mathcal{O}\left(n^{2}\right)$ to $\mathcal{O}(n)$. However, in this thesis this is implemented for all the node displacements since the required nodes for ultrasound synthesis are hard to predetermine and the positions of some element corners are also needed at haptic rates (e.g., the corners close to the needle tip are used for element penetration check described in Section 2.5).

The above mentioned method is half of the condensation approach by Bro-Nielsen [89]. The other half suggests using only the columns of $\boldsymbol{K}^{-1}$ which correspond to the applied external forces on the system. The fact that external forces on the tissue mesh exist only at the tissue mesh nodes, which are constrained on the needle, led to the exploitation of that other half. These nodes are called the contact nodes or the working nodes. In other words, while the needle base is being manipulated, the interaction with the tissue model only occurs at the contact nodes, which is a subset of tissue mesh vertices lying on the needle. Thus, if (2.1) is reordered so that contact nodes are at the end, the formulation becomes:

$$
\begin{align*}
& \Rightarrow \boldsymbol{u}=\left[\begin{array}{ll}
\left(\boldsymbol{K}^{-1}\right)_{00} & \left(\boldsymbol{K}^{-1}\right)_{01} \\
\left(\boldsymbol{K}^{-1}\right)_{10} & \left(\boldsymbol{K}^{-1}\right)_{11}
\end{array}\right]\left[\begin{array}{l}
\boldsymbol{f}_{0} \\
\boldsymbol{f}_{1}
\end{array}\right] \\
\boldsymbol{f}_{0}=\mathbf{0} & \Rightarrow \boldsymbol{u}=\left[\begin{array}{l}
\left(\boldsymbol{K}^{-1}\right)_{01} \\
\left(\boldsymbol{K}^{-1}\right)_{11}
\end{array}\right] \boldsymbol{f}_{1} \\
& \Rightarrow\left\{\begin{array}{l}
\boldsymbol{u}_{0}=\left(\boldsymbol{K}^{-1}\right)_{01} \boldsymbol{f}_{1} \\
\boldsymbol{u}_{1}=\left(\boldsymbol{K}^{-1}\right)_{11} \boldsymbol{f}_{1}
\end{array}\right. \tag{2.2}
\end{align*}
$$

where for the vectors $\boldsymbol{u}$ and $\boldsymbol{f}$ the subscripts 1 and 0 denote the rows corresponding to the contact nodes and the rest of the nodes, respectively.

### 2.1.2 Condensed Subsystem

The solution to (2.2) requires the non-zero forces, $\boldsymbol{f}_{1}$, on the contact nodes. During the simulation these contact boundary conditions are set by the needle. However, they are mostly applied as pure displacement boundary conditions or as mixed displacement-force boundary conditions (these cases correspond to the stick and the slip states, respectively, as described in the following section). Fortunately, the entire relationship between the displacements $\boldsymbol{u}_{1}$ and the forces $\boldsymbol{f}_{1}$ of the contact nodes can be described by the set of equations on the last line of (2.2). This smaller system of equations is called the condensed subsystem. Its matrix representation $\left(\boldsymbol{K}^{-1}\right)_{11}$, which is a submatrix of the inverted global stiffness matrix $\boldsymbol{K}^{-1}$, is named the condensed (system) matrix and denoted by $\boldsymbol{K}_{H}$ basically to refer further in the implementation chapter. Note that this definition attributes only to a special (the initial - before any update) form of the working matrix that will be defined later in this chapter.

As an example, when all contact nodes are stuck, there will be displacement constraints on all of them and the condensed subsystem is then expressed as:

$$
\begin{equation*}
\boldsymbol{f}_{1}=\boldsymbol{K}_{H}^{-1} \boldsymbol{u}_{1} . \tag{2.3}
\end{equation*}
$$

Here if the number of working nodes is $w$, the dimension of the system will become $3 w$. This set of equations, consequently its matrix definition, need to be updated at each iteration according to the contact node states (Section 2.3) and needle orientation (Section 2.4).

### 2.2 Contact Behaviour

After the successful implementation of the stick-slip approach by DiMaio [40], we integrated the same method into our simulator. It had been proven to be computationally fast and yet visually realistic. It is based on physics and makes use of the fact that two surfaces start slipping only if the interaction force exceeds their friction and they continue slipping as long as they are in motion with respect to each other. This shapes the quasi-static behaviour of our simulation. At an iteration, a node either slips along or sticks onto the needle shaft. This decision for each contact node is made at every iteration depending on its current state. Our stick-slip automaton is shown in Figure 2.1. Here $F_{s}$ is the static friction threshold at the needle-tissue interface and $F_{\text {ext }}$ is the magnitude of the axial force exerted by the contact node on the needle surface, and $v_{i}$ and $v_{i-1}$ are the velocities in the current and previous iterations of a simulation, respectively. So, the sign of $v_{i} \cdot v_{i-1}$ shows whether the direction of the motion has changed since the previous iteration. A direction switch implies a zero-crossing of the velocity which consequently denotes the end of slipping due to the above mentioned kinematic fact.

After being constrained onto the needle, a contact node stays on it until it slips off the tip. Therefore, its lateral displacement with respect to the local orientation of the needle is set by the needle shaft. On the other hand, along the shaft axis the contact is switched between force and


Figure 2.1: Automaton describing the stick-slip friction model of a contact node.
displacement constraints such that in the stick state the displacement is set by the needle, and in the slip state an axial friction force is exerted by the shaft. These axial constraint switches are applied using the boundary condition change technique described in Section 2.3, where the local axis along the shaft is chosen to be $y$ in our particular implementation shown in Figure 2.2.


Figure 2.2: Tissue and needle-base axes in our implementation.

Since the unknowns in (2.3) switch between displacement and force, depending on the boundary condition, this system of equations can be rearranged in order to have all the unknown axis components on the left-hand side as:

$$
\begin{equation*}
\underline{\boldsymbol{x}}=\boldsymbol{K}_{W} \underline{\boldsymbol{y}} \tag{2.4}
\end{equation*}
$$

where $\boldsymbol{K}_{W}$ is the working matrix, which is the boundary condition changed and local frame adjusted version of the condensed matrix $\boldsymbol{K}_{H}$ at any simulation iteration. In order to correctly define a relationship between displacements and forces of the contact nodes, the matrix needs to be updated if there is a stick-slip state change at an iteration. For example, at any given time if the contact node at the tip is stuck while two other contact nodes are slipping along the shaft, $\underline{\boldsymbol{x}}$ and $\underline{\boldsymbol{y}}$ should look like:

$$
\begin{align*}
& \underline{\boldsymbol{x}}=\left[\begin{array}{lllllllll}
f_{1}^{x} & u_{1}^{y} & f_{1}^{z} & f_{2}^{x} & u_{2}^{y} & f_{2}^{z} & f_{3}^{x} & f_{3}^{y} & f_{3}^{z}
\end{array}\right]^{T}  \tag{2.5}\\
& \underline{\boldsymbol{y}}=\left[\begin{array}{lllllllll}
u_{1}^{x} & f_{1}^{y} & u_{1}^{z} & u_{2}^{x} & f_{2}^{y} & u_{2}^{z} & u_{3}^{x} & u_{3}^{y} & u_{3}^{z}
\end{array}\right]^{T}
\end{align*}
$$

The above scheme necessitates the constraint application on a local frame that is axially oriented with the needle shaft at the contact points. For the three contact nodes given above, an instance of local frames on a bent needle can be oriented as in Figure 2.3. When the needle base is manipulated and/or the needle shaft flexes, these local frames also need to be updated (rotated) accordingly as described in Section 2.4.


Figure 2.3: An instance of contact node local frames lying on a bent needle.

### 2.3 Boundary Condition Change

When a contact state changes, in (2.4) one row of the knowns $\underline{\boldsymbol{y}}$ has to be swapped with the corresponding row of the unknowns $\underline{\boldsymbol{x}}$ in order to accommodate the new boundary condition. The procedure for this operation is easier to show on a sample case. Nonetheless, note that the following operations are applied without loss of generality. So, this sample case can later be generalized.

First let us assume that all the contact nodes are stuck. Thus, the condensed subsystem is denoted as:

Here a generic variable $r$ is used just because this same technique is utilized also for mesh adaptation with a full-rank matrix. So, the method and the related appendix can be applied to this other case using a proper $r$ value. In here $r$ is obviously $3 w$, where $w$ is the number of working nodes.

For any given node axis, whenever the force is known and the displacement is sought, the set
of linear equations to be solved becomes:

$$
\left[\begin{array}{c}
f_{1}  \tag{2.7}\\
\vdots \\
f_{i-1} \\
u_{i} \\
f_{i+1} \\
\vdots \\
f_{r}
\end{array}\right]=\boldsymbol{K}_{W}^{\prime}\left[\begin{array}{c}
u_{1} \\
\vdots \\
u_{i-1} \\
f_{i} \\
u_{i+1} \\
\vdots \\
u_{r}
\end{array}\right]
$$

This updated working subsystem which is denoted by this new matrix $\boldsymbol{K}_{W}{ }^{\prime}$ consequently has its unknowns in the vector on the left, which is the new $\underline{\boldsymbol{x}}$, and the known variables on the right, which is the new $\underline{\boldsymbol{y}}$ vector. In Appendix B, $\boldsymbol{K}_{W}{ }^{\prime}$ is proven to be of the following form:

$$
\begin{equation*}
\boldsymbol{K}_{W}^{\prime}=\boldsymbol{K}_{W}-\frac{1}{k_{(i, i)}} \boldsymbol{c}_{i} \boldsymbol{r}_{i} \tag{2.8}
\end{equation*}
$$

where $\boldsymbol{c}_{i}$ and $\boldsymbol{r}_{i}$ are the modified forms of the $i^{\text {th }}$ column and $i^{\text {th }}$ row of $\boldsymbol{K}^{w}$, respectively, such that:

$$
\begin{align*}
& \boldsymbol{c}_{i}=\left[k_{(1, i)}, \ldots, k_{(i-1, i)},\left(k_{(i, i)}+1\right), k_{(i+1, i)}, \ldots, k_{(r, i)}\right]^{T}  \tag{2.9}\\
& \boldsymbol{r}_{i}=\left[k_{(i, 1)}, \ldots, k_{(i, i-1)},\left(k_{(i, i)}-1\right), k_{(i, i+1)}, \ldots, k_{(i, r)}\right]
\end{align*}
$$

So, via an inexpensive low-rank update, the boundary condition of a node for a given axis can be switched between force and displacement. This is similar to the capacitance matrix strategy of James and Pai in [28] and is also implemented in [40].

The $\boldsymbol{c}_{i} \boldsymbol{r}_{i}$ multiplication results in $\mathcal{O}\left(w^{2}\right)$ computations for this operation. Throughout this thesis, computations refer only to multiplications and divisions.
$\mathcal{O}\left(w^{2}\right)$ makes the speed of this algorithm greatly dependant on the number of contact nodes. Although this is a relatively fast operation, its time requirement is not constant per iteration. Its biggest computational burden comes as one boundary change per contact node, when the needle penetration/retraction stops so all contact nodes switch from slip to stick state in a single iteration. Consequently, this sets a limit on how coarse the mesh can be. In the current implementation and meshing, there are usually no more than 10 nodes on the needle.

### 2.4 Local Frame Rotation

Although this section is based on the work by DiMaio [40], the increase in matrix dimensions for 3D needed some modifications on the method, especially for the slip state, which required a new algebraic solution. Furthermore, the need for more than one angle in 3D resulted in extra steps. For example, whereas in 2D an angle-wise addition of two consecutive rotations can solve
the total rotation, in 3D consecutive rotations needed transformation matrices to be employed.
The local frames of the contact nodes seen in Figure 2.3 rotated as the needle twists or flexes. Depending on its state, this rotation is handled correspondingly.

### 2.4.1 Stick State

For a node $i$ in this state, all three axes are under displacement constraints. So, the vectors ${ }^{0} \boldsymbol{u}_{i}$ and ${ }^{0} \boldsymbol{f}_{i}$ in the initial coordinate frame can be expressed as follows:

$$
\begin{align*}
& {\left[\begin{array}{c}
{ }^{0} u_{i}^{x} \\
{ }^{0} u_{i}^{y} \\
{ }^{0} u_{i}^{z}
\end{array}\right]=\left[\begin{array}{lll}
R_{11} & R_{12} & R_{13} \\
R_{21} & R_{22} & R_{23} \\
R_{31} & R_{32} & R_{33}
\end{array}\right]\left[\begin{array}{c}
{ }^{1} u_{i}^{x} \\
{ }^{1} u_{i}^{y} \\
{ }^{1} u_{i}^{z}
\end{array}\right]}  \tag{2.10}\\
& {\left[\begin{array}{c}
0 f_{i}^{x} \\
{ }^{0} f_{i}^{y} \\
{ }^{0} f_{i}^{z}
\end{array}\right]=\left[\begin{array}{lll}
R_{11} & R_{12} & R_{13} \\
R_{21} & R_{22} & R_{23} \\
R_{31} & R_{32} & R_{33}
\end{array}\right]\left[\begin{array}{c}
1 f_{i}^{x} \\
{ }^{1} f_{i}^{y} \\
{ }^{1} f_{i}^{z}
\end{array}\right]}
\end{align*}
$$

where ${ }^{1} \boldsymbol{u}_{i}$ and ${ }^{1} \boldsymbol{f}_{i}$ are the vectors after rotating the coordinate frame and the matrix $R_{3 \times 3}$ is the transformation from the rotated frame to the initial frame. This expression is chosen for simplicity in the equations below. (2.4) can then be rewritten in the rotated frame as:

$$
\begin{align*}
& { }^{0} \underline{\boldsymbol{x}}=\boldsymbol{K}_{W}{ }^{0} \underline{\boldsymbol{y}} \\
& \Rightarrow \quad{ }^{1} \underline{\boldsymbol{x}}=\left(A^{T} \boldsymbol{K}_{W} A\right)^{1} \underline{\boldsymbol{y}}  \tag{2.11}\\
& \Rightarrow \quad{ }^{1} \underline{\boldsymbol{x}}=\boldsymbol{K}_{W}{ }^{\prime}{ }^{1} \underline{y}
\end{align*}
$$

where $\boldsymbol{K}_{W}{ }^{\prime}$ is the updated working matrix and:

$$
A=\left[\begin{array}{cccccccc}
1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0  \tag{2.12}\\
0 & 1 & & 0 & 0 & 0 & & 0 \\
\vdots & & \ddots & & & & & \vdots \\
0 & 0 & & R_{11} & R_{12} & R_{13} & & 0 \\
0 & 0 & & R_{21} & R_{22} & R_{23} & & 0 \\
0 & 0 & & R_{31} & R_{32} & R_{33} & & 0 \\
\vdots & & & & & & \ddots & \vdots \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1
\end{array}\right]
$$

If the rotation angles $(\phi, \theta, \psi)$ are given in roll-pitch-yaw form, then $R$ becomes:

$$
R=\left[\begin{array}{lll}
R_{11} & R_{12} & R_{13}  \tag{2.13}\\
R_{21} & R_{22} & R_{23} \\
R_{31} & R_{32} & R_{33}
\end{array}\right]=\left[\begin{array}{ccc}
c \phi c \theta & c \phi s \theta s \psi-s \phi c \psi & c \phi s \theta c \psi+s \phi s \psi \\
s \phi c \theta & s \phi s \theta s \psi+c \phi c \psi & s \phi s \theta c \psi-c \phi s \psi \\
-s \theta & c \theta s \psi & c \theta c \psi
\end{array}\right]
$$

where $s$ and $c$ denotes $\sin$ and $\cos$ functions, respectively. This is an $\mathcal{O}(1)$ update per contact node in every iteration.

### 2.4.2 Slip State

While a contact node is slipping, the constraint along the shaft, which is the local $y$ axis, becomes a friction force that has a magnitude equal to the needle-tissue slippage. Thus, from (2.4) the mixed force-displacement constraints are written in the form:

$$
\left[\begin{array}{c}
0_{x_{1}}  \tag{2.14}\\
\vdots \\
0_{f}^{x} \\
0_{i} \\
0_{i}^{y} \\
0_{f_{i}^{z}} \\
\vdots \\
{ }_{0} x_{3 w}
\end{array}\right]=\boldsymbol{K}_{W}\left[\begin{array}{c}
0_{y_{1}} \\
\vdots \\
0_{u} \\
0_{i}^{x} \\
0_{i}^{y} \\
0_{u_{i}^{z}} \\
\vdots \\
0_{y_{3 w}}
\end{array}\right]
$$

which can then be separated into:

$$
\left[\begin{array}{c}
{ }^{0} x_{1}  \tag{2.15}\\
\vdots \\
{ }^{0} f_{i}^{x} \\
0 \\
{ }^{0} f_{i}^{z} \\
\vdots \\
{ }^{0} x_{3 w}
\end{array}\right]+\left[\begin{array}{c}
0 \\
\vdots \\
0 \\
{ }^{0} u_{i}^{y} \\
0 \\
\vdots \\
0
\end{array}\right]=\boldsymbol{K}_{W}\left(\left[\begin{array}{c}
0 \\
\vdots \\
0 \\
{ }^{0} f_{i}^{y} \\
0 \\
\vdots \\
0
\end{array}\right]+\left[\begin{array}{c}
0 y_{1} \\
\vdots \\
{ }^{0} u_{i}^{x} \\
0 \\
{ }^{0} u_{i}^{z} \\
\vdots \\
{ }^{y_{3 w}}
\end{array}\right]\right)
$$

Using (2.10), each term above can be denoted in terms of the rotated frame vectors ${ }^{1} \underline{\boldsymbol{x}}$ and ${ }^{1} \underline{y}:$

$$
\begin{equation*}
M^{1} \underline{\boldsymbol{x}}+N^{1} \underline{\boldsymbol{y}}=\boldsymbol{K}_{W}\left[N^{1} \underline{\boldsymbol{x}}+M^{1} \underline{\boldsymbol{y}}\right] \tag{2.16}
\end{equation*}
$$

where $M$ and $N$ are sparse transformation matrices of the form:

$$
\begin{align*}
& M=\left[\begin{array}{cccccccc}
1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
0 & 1 & & 0 & 0 & 0 & & 0 \\
\vdots & & \ddots & & & & & \vdots \\
0 & 0 & & R_{11} & 0 & R_{13} & & 0 \\
0 & 0 & & 0 & R_{22} & 0 & & 0 \\
0 & 0 & & R_{31} & 0 & R_{33} & & 0 \\
\vdots & & & & & & \ddots & \vdots \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1
\end{array}\right]  \tag{2.17}\\
& N=\left[\begin{array}{cccccccc}
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
0 & 0 & & 0 & 0 & 0 & & 0 \\
\vdots & & \ddots & & & & & \vdots \\
0 & 0 & & 0 & R_{12} & 0 & & 0 \\
0 & 0 & & R_{21} & 0 & R_{23} & & 0 \\
0 & 0 & & 0 & R_{32} & 0 & & 0 \\
\vdots & & & & & \ddots & \vdots \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0
\end{array}\right] \tag{2.18}
\end{align*}
$$

(2.16) leads to the expression of ${ }^{1} \underline{\boldsymbol{x}}$ in terms of $\underline{\boldsymbol{y}}^{\boldsymbol{y}}$ as:

$$
\begin{align*}
{ }^{1} \underline{\boldsymbol{x}} & =\left[M-\boldsymbol{K}_{W} N\right]^{-1}\left[\boldsymbol{K}_{W} M-N\right]^{1} \underline{\boldsymbol{y}}  \tag{2.19}\\
\Rightarrow{ }^{1} \underline{\boldsymbol{x}} & =\boldsymbol{K}_{W}{ }^{\prime}{ }^{1} \underline{\boldsymbol{y}}
\end{align*}
$$

which gives the updated working matrix $\boldsymbol{K}_{W}{ }^{\prime}$.
The implementation of this formula requires the inversion of $\left[M-\boldsymbol{K}^{w} N\right]$. Since it is sparse as shown below, its inverse can be computed symbolically and then be hard-coded in the C file as an algebraic set of equations. $N$ has only 4 non-zero values, so $\boldsymbol{K}_{3 w \times 3 w}^{w} N_{3 w \times 3 w}$ takes the form $\left[\emptyset_{3 w \times 3(j-1)} X_{3 w \times 3} \emptyset_{3 w \times 3(w-j)}\right]$ that has 3 non-zero columns, where $j$ is the working node to be
updated. Thus, the matrix to be inverted is expressed as:

$$
M-\boldsymbol{K}^{w} N=\left[\begin{array}{cccccccc}
1 & 0 & \cdots & a_{(1,3 j-2)} & a_{(1,3 j-1)} & a_{(1,3 j)} & \cdots & 0  \tag{2.20}\\
0 & 1 & \cdots & a_{(2,3 j-2)} & a_{(2,3 j-1)} & a_{(2,3 j)} & \cdots & 0 \\
\vdots & & \ddots & \vdots & \vdots & \vdots & & \vdots \\
0 & 0 & \cdots & a_{(3 j-2,3 j-2)} & a_{(3 j-2,3 j-1)} & a_{(3 j-2,3 j)} & \cdots & 0 \\
0 & 0 & \cdots & a_{(3 j-1,3 j-2)} & a_{(3 j-1,3 j-1)} & a_{(3 j-1,3 j)} & \cdots & 0 \\
0 & 0 & \cdots & a_{(3 j, 3 j-2)} & a_{(3 j, 3 j-1)} & a_{(3 j, 3 j)} & \cdots & 0 \\
\vdots & & & \vdots & \vdots & \vdots & \cdots & \vdots \\
0 & 0 & \cdots & a_{(3 w, 3 j-2)} & a_{(3 w, 3 j-1)} & a_{(3 w, 3 j)} & \cdots & 1
\end{array}\right] .
$$

For convenience, elements of the $3 \times 3$ central matrix will be renamed as:

$$
\left[\begin{array}{lll}
a_{11} & a_{12} & a_{13}  \tag{2.21}\\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{array}\right]=\left[\begin{array}{ccc}
a_{(3 j-2,3 j-2)} & a_{(3 j-2,3 j-1)} & a_{(3 j-2,3 j)} \\
a_{(3 j-1,3 j-2)} & a_{(3 j-1,3 j-1)} & a_{(3 j-1,3 j)} \\
a_{(3 j, 3 j-2)} & a_{(3 j, 3 j-1)} & a_{(3 j, 3 j)}
\end{array}\right]
$$

Symbolic toolbox of Matlab ${ }^{1}$ gives the inverted matrix in the following form:

$$
\left[M-\boldsymbol{K}^{w} N\right]^{-1}=\left[\begin{array}{cccccccc}
1 & 0 & \cdots & a_{(1,3 j-2)}^{\prime} & a_{(1,3 j-1)}^{\prime} & a_{(1,3 j)}^{\prime} & \cdots & 0  \tag{2.22}\\
0 & 1 & \cdots & a_{(2,3 j-2)}^{\prime} & a_{(2,3 j-1)}^{\prime} & a_{(2,3 j)}^{\prime} & \cdots & 0 \\
\vdots & & \ddots & \vdots & \vdots & \vdots & & \vdots \\
0 & 0 & \cdots & a_{11}^{\prime} & a_{12}^{\prime} & a_{13}^{\prime} & \cdots & 0 \\
0 & 0 & \cdots & a_{21}^{\prime} & a_{22}^{\prime} & a_{23}^{\prime} & \cdots & 0 \\
0 & 0 & \cdots & a_{31}^{\prime} & a_{32}^{\prime} & a_{33}^{\prime} & \cdots & 0 \\
\vdots & & & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & a_{(3 w, 3 j-2)}^{\prime} & a_{(3 w, 3 j-1)}^{\prime} & a_{(3 w, 3 j)}^{\prime} & \cdots & 1
\end{array}\right]
$$

where the elements are:

$$
\begin{align*}
& \text { for the rows } \\
& 3 j-2,3 j-1,3 j: \\
& a_{11}^{\prime}=-c_{11} \quad a_{12}^{\prime}=-c_{21} \quad a_{13}^{\prime}=-c_{31} \\
& a_{21}^{\prime}=-c_{12} \quad a_{22}^{\prime}=-c_{22} \quad a_{23}^{\prime}=-c_{32} \\
& a_{31}^{\prime}=-c_{13} \quad a_{32}^{\prime}=-c_{23} \quad a_{33}^{\prime}=-c_{33} \\
& \begin{array}{l}
\text { for the rest of the rows } \\
\forall_{i \in\{1, \ldots, 3 j-3,3 j+1, \ldots, 3 w\}}:
\end{array} \begin{cases}a_{(3 i, j)}^{\prime} & =c_{11} a_{(3 i, j)}+c_{12} a_{(3 i, j)}+c_{13} a_{(3 i, j)} \\
a_{(3 i+1, j)}^{\prime} & =c_{21} a_{(3 i+1, j)}+c_{22} a_{(3 i+1, j)}+c_{23} a_{(3 i+1, j)} \\
a_{(3 i+2, j)}^{\prime} & =c_{31} a_{(3 i+2, j)}+c_{32} a_{(3 i+2, j)}+c_{33} a_{(3 i+2, j)}\end{cases} \tag{2.23}
\end{align*}
$$

[^4]where the $c$ values are found in the following form:
\[

$$
\begin{array}{lll}
c_{11}=\frac{a_{23} a_{32}-a_{22} a_{33}}{r} & c_{12}=\frac{a_{33} a_{21}-a_{23} a_{31}}{r} & c_{13}=\frac{a_{22} a_{31}-a_{32} a_{21}}{r} \\
c_{21}=\frac{a_{33} a_{12}-a_{13} a_{32}}{r} & c_{22}=\frac{a_{13} a_{31}-a_{11} a_{33}}{r} & c_{23}=\frac{a_{11} a_{32}-a_{31} a_{12}}{r}  \tag{2.24}\\
c_{31}=\frac{a_{13} a_{22}-a_{12} a_{23}}{r} & c_{32}=\frac{a_{23} a_{11}-a_{13} a_{21}}{r} & c_{33}=\frac{a_{12} a_{21}-a_{22} a_{11}}{r}
\end{array}
$$
\]

while $r=a_{13} a_{21} a_{32}-a_{13} a_{22} a_{31}-a_{23} a_{11} a_{32}-a_{33} a_{12} a_{21}+a_{22} a_{33} a_{11}+a_{31} a_{12} a_{23}$.
In every iteration, this update requires $\mathcal{O}(w)$ computations per slipping node.

### 2.5 Decision of Remeshing

Due to the simulation design and implementation, the manipulated tissue nodes can only be on the needle. To relax this restriction a virtual coupling could have been achieved by interpolating the forces of the surrounding nodes and linearly distributing the needle displacements onto them similarly to the approach of Berkley et al. [39]. However, this would make the application of boundary constraints difficult. Also a larger working matrix would then be necessary which would make the boundary condition changes slower. Thus, the approach of constraining the tissue nodes onto the needle is preferred.

Alterovitz et al. [9] and DiMaio [40] use similar techniques as parts of their applications. In brief, as the needle is advancing when its tip enters into a new tissue element, the closest corner of this newly intercepted element is rediscretized in order to have a mesh node coinciding with the needle. Figure 2.4 depicts a 2 D case. Note that all the element geometry shown in here is from a deformed mesh. Although that geometry's nominal structure is also referred in this section, it is not drawn for representation simplicity.

While the needle is advancing in the tissue, if its tip penetrates into a "new" (having none of its corners already on the needle) element as in Figure $2.4(\mathrm{~b})$, then the node addition steps are taken as follows:

1. The point that $P$ should move to is determined (it is $P^{\prime}$ in our implementation).
2. This point (in ours, $P^{\prime}$ ) is mapped to the nominal mesh.
3. The region is remeshed as described in mesh modification (Section 2.6): The nominal position of $P$ is moved to the nominal position of the desired point that is found in step 2.
4. The condensed matrix is updated to accommodate this new working node.
5. Along with the other working node constraints, this node is also constrained on the needle (in our implementation, confined to the tip at stuck state initially).

Although from a general perspective, not all the intercepted elements need to have a corner as a working node (let us say, some of the nodes could be skipped), this was not preferred since


Figure 2.4: When a new element is punctured, the subregion is remeshed in order to have a node around the needle tip. In (c), solid lines represent the new elements to be formed and the dashed lines denote the elements to be removed during remeshing.
more nodes would achieve a tighter coupling between tissue and the needle. Moreover, the above mentioned approach of adding all the nodes leads to some useful assumptions (such as "the tip can only be in an element connected to the last working node on the shaft") which can speed up other algorithms throughout the implementation. The tip penetration check is performed using the Algorithm 4.2 in Section 4.4. Each element connected to the last contact node is checked for tip inclusion.

For step 1 of node addition, [40] preferred the point of edge intersection, $P^{\prime \prime}$ in Figure 2.4. First this was implemented in 3D as well. It required plane-line intersection calculations, as opposed to the line-line case of 2D. Later this was changed to $P^{\prime}$ considering this node is to be stuck at the tip in step 5 . This also removed the intersection calculations.

In step 2, nominal position calculation is easily performed by [40] in 2D for $P^{\prime \prime}$. The ratio of $\left|P P^{\prime \prime}\right|$ to $\left|P P_{2}\right|$ is applied on the nominal mesh. However, the 3D case required a slightly longer analytical solution. The solution for an inner node, $P^{\prime}$ was even harder analytically. Thus, the author made use of the basis functions to correctly map this point to the nominal mesh in fewer well-structured calculations. First the basis function from the deformed element $\left(P P_{2} P_{3}\right)$ to its nominal geometry is found, and then using this basis function $P^{\prime}$ is identified in the nominal mesh. This is the same method (Algorithm 4.3) applied for the pixel positions of the ultrasound imaging plane in Section 4.5.

Unfortunately, using flexible needle the above scheme caused instabilities when constraining the tip to either $P^{\prime}$ or $P^{\prime \prime}$. After confining the node on the needle in step 5 , there has been a residual force on that node of the mesh, which only settled in many iterations, if it converged at all. Further investigation revealed the fact that finding the exact point which would deform onto $P^{\prime}$ is theoretically impossible. When a mesh node is relocated, since the global stiffness matrix changes, even the point $P$ will not be at its previous location with the same constraints applied on the contact nodes. This is due to the different response of the new tissue discretization. Nevertheless, preventing this instability caused by the slight position miscalculation is only a
matter of using a higher damping for the tissue-needle coupling described in Section 2.7. Briefly, since both models are elastic, their coupled system has a tendency to resonate and it needs to damped. The above mentioned tip residual force is a case where this coupled system is pushed to its stability limits. Therefore, the damping was determined empirically where the needle does not resonate too much during insertion but still responds quickly.

### 2.6 Mesh Modification

Since the global stiffness matrix $\boldsymbol{K}$ is obtained by summing up all individual element stiffness matrices $\left(K^{e}\right)$, subtracting them will effectively remove the elements' stiffnesses from this global matrix.

For example, consider a part of the tissue that is formed by the elements $e_{1}=P_{2} P_{4} P_{3} P$, $e_{2}=P_{1} P_{3} P_{4} P, e_{3}=P_{1} P_{4} P_{2} P$, and $e_{4}=P_{1} P_{2} P_{3} P$ as seen in Figure 2.5. Removing those four elements from the global stiffness matrix $\boldsymbol{K}$ decouples the node $P$ from the rest of the tissue. In other words, it inhibits all the strain effects caused by $P$ on other nodes and caused on $P$ by other nodes. Consequently, we can safely relocate it to anywhere (let us say $P^{\prime}$ ) inside the tetrahedron $P_{1} P_{2} P_{3} P_{4}$ and add the new tissue elements $e_{1}^{\prime}=P_{2} P_{4} P_{3} P^{\prime}, e_{2}^{\prime}=P_{1} P_{3} P_{4} P^{\prime}, e_{3}^{\prime}=P_{1} P_{4} P_{2} P^{\prime}$, and $e_{4}{ }^{\prime}=P_{1} P_{2} P_{3} P^{\prime}$ onto the global stiffness matrix $\boldsymbol{K}$ to have an intact mesh. Here, note that in most mesh geometries the nodes usually have more than four neighbouring elements. Yet, the descriptions below are made without loss of generality, so they are independent from the number of surrounding elements.


Figure 2.5: Four elements neighbouring to node $P$ which is to be repositioned.

Thus, the general form of the matrix update for a displacement $\boldsymbol{u}$ of a node $P$ which relocates it to the position $P^{\prime}=P+\boldsymbol{u}$ is written as follows:

$$
\begin{equation*}
\boldsymbol{K}^{\prime}=\boldsymbol{K}-\sum_{e_{i}} \operatorname{global}\left(K^{e_{i}}(P)\right)+\sum_{e_{i}} \operatorname{global}\left(K^{e_{i}}\left(P^{\prime}\right)\right) \tag{2.25}
\end{equation*}
$$

where $e_{i}$ are the elements connected to $P$. The global function maps the element matrices to
the global matrix numbering (Appendix A.5). Note that this modifies only a submatrix of $\boldsymbol{K}$ corresponding to the corner vertices of elements $e_{i}$. For simplicity, throughout the rest of this section, let us assume that the matrices are reordered in order to have the nodes that are directly connected to $P$ at the end and call those nodes $P_{i}$. Let $m$ be the number of such nodes (namely $P$ 's valence or order).

### 2.6.1 $\mathcal{O}\left(m n^{2}\right)$ Approach

The real-time code uses the inverse stiffness matrix $\boldsymbol{K}^{-1}$ as in (2.1). However, $K^{e}$ defines a relationship from the displacements to the forces. Therefore, a direct operation on this $\boldsymbol{K}^{-1}$ is not possible. One approach can apply the operations of (2.25) on the original global stiffness matrix $\boldsymbol{K}$ and then invert the entire full-rank matrix. As discussed earlier, a real-time inversion of rank $3 n$ is infeasible. A fast approach is the extended version of Sherman-Morrison formula, namely the Woodbury formula [90]. If $\Delta \mathrm{K}=-\sum_{e_{i}} K^{e_{i}}(P)+\sum_{e_{i}} K^{e_{i}}\left(P^{\prime}\right)$, then the inversion of the perturbed matrix is given by:

$$
\begin{equation*}
\left(\boldsymbol{K}+V \Delta \mathrm{~K} V^{T}\right)^{-1}=\boldsymbol{K}^{-1}-\left[\boldsymbol{K}^{-1} V \Delta \mathrm{~K}\left(1+V^{T} \boldsymbol{K}^{-1} V \Delta \mathrm{~K}\right)^{-1} V^{T} \boldsymbol{K}^{-1}\right] \tag{2.26}
\end{equation*}
$$

Here $V$ is a $3 n \times 3 m$ matrix having the last row as an identity matrix of the form $I_{3 m \times 3 m}$. This requires $27 n m(n+2 m)$ computations, in addition to a $3 m \times 3 m$ matrix inversion which can be done in $\mathcal{O}\left(m^{3}\right)$ in the worst case. Although $n \gg m$ is a valid assumption, $m$ will also be included in the big $\mathcal{O}$ notation to express the effect of node valence. Instead of this $\mathcal{O}\left(m n^{2}\right)$ approach, DiMaio [40] employed another $\mathcal{O}\left(m n^{2}\right)$ method with slightly more computations ( $162 m n^{2}$ ). He converts a part of $\boldsymbol{K}^{-1}$ to displacement constraints, and after applying the $K^{e_{i}}$ operations on this region, converts it back. The conversion is done by swapping the 3 m rows (corresponding to $P_{i}$ ) at the end using boundary switches in Section 2.3. That yields to:

$$
\left[\begin{array}{c}
\boldsymbol{u}_{P_{1}}  \tag{2.27}\\
\vdots \\
\boldsymbol{u}_{P_{n-m}} \\
\hline \boldsymbol{f}_{P_{n-m+1}} \\
\vdots \\
\boldsymbol{f}_{P_{n}}
\end{array}\right]=\left[\begin{array}{l|l}
\boldsymbol{L}_{11} & \boldsymbol{L}_{12} \\
\hline \boldsymbol{L}_{21} & \boldsymbol{L}_{22}
\end{array}\right]\left[\begin{array}{c}
\boldsymbol{f}_{P_{1}} \\
\vdots \\
\boldsymbol{f}_{P_{n-m}} \\
\hline \boldsymbol{u}_{P_{n-m+1}} \\
\vdots \\
\boldsymbol{u}_{P_{n}}
\end{array}\right]
$$

Here $\boldsymbol{u}$ and $\boldsymbol{f}$ values are triplets containing all three axes. Naming the vectors conveniently gives:

$$
\left[\begin{array}{c}
\boldsymbol{U}_{1}  \tag{2.28}\\
\hline \boldsymbol{F}_{2}
\end{array}\right]=\left[\begin{array}{l|l}
\boldsymbol{L}_{11} & \boldsymbol{L}_{12} \\
\hline \boldsymbol{L}_{21} & \boldsymbol{L}_{22}
\end{array}\right]\left[\begin{array}{l}
\boldsymbol{F}_{1} \\
\hline \boldsymbol{U}_{2}
\end{array}\right]
$$

Note that none of the submatrices $\boldsymbol{L}$ holds values of either $\boldsymbol{K}$ or $\boldsymbol{K}^{-1}$ directly. Instead they
define a mixed boundary condition similar to (2.4) such that:

$$
\begin{align*}
\boldsymbol{U}_{1} & =\boldsymbol{L}_{11} \boldsymbol{F}_{1}+\boldsymbol{L}_{12} \boldsymbol{U}_{2} \\
\boldsymbol{F}_{2} & =\boldsymbol{L}_{21} \boldsymbol{F}_{1}+\boldsymbol{L}_{22} \boldsymbol{U}_{2} \tag{2.29}
\end{align*}
$$

The advantage of this conversion is that now $K^{e_{i}}$ subtractions and additions can be applied on $\boldsymbol{L}_{22}$, since all the relationships of the displacements to the forces of the nodes $P_{i}$ are now imbedded in it. Thus, finally subtracting the previous elements' stiffnesses and adding the new elements' stiffnesses, and then applying the boundary condition switches once more onto those nodes (to convert them back to original force constraints) results in the $\left(\boldsymbol{K}^{-1}\right)^{\prime}$, the inverse global stiffness matrix of the new mesh. Note that finding and adding/subtracting the individual element stiffness matrices, $K^{e_{i}}$, take negligible time compared to the other operations.

### 2.6.2 $\mathcal{O}\left(m^{2} n\right)$ Approach

We propose an approximation to this matrix update in $\mathcal{O}\left(m^{2} n\right)$. Before describing this, let us examine how the remeshing of a node $P$ can affect the inverse global stiffness matrix $\boldsymbol{K}^{-1}$. Firstly, changing the position of $P$ with respect to the other nodes will alter two factors, the displacement caused on $P$ by other nodes' forces and the displacements of other nodes caused by the forces on $P$. In other words, a node's position in the mesh definitely determines the values of the columns and rows of $\boldsymbol{K}^{-1}$ corresponding to $P$. Note that if the tissue deformation was not modeled using discrete elements, this effect of $P$ on itself would be the only required change for a position update. However, since the effects diffuse in finite element model through elements, surrounding nodes of $P$ affect each other also through these elemenets. For instance considering two neighbours of $P$ that are $P_{1}$ and $P_{2}$, the displacement of $P_{2}$ is also affected from a force on $P_{1}$ through the element $P P_{1} P_{2}$ so that the position of $P$ can affect their mutual behaviour. We will call these the secondary effects. In fact, there are tertiary effects and so on, meaning that even the most distal nodes reciprocal behaviour is slightly affected by where exactly this node is located. As mentioned earlier, all effects except for the primary are side-effects of FEM. Obviously in reality, the mutual behaviour of the distal nodes should have nothing to do with the virtual partitioning of the in-between material.

To see this, we analyzed an instance of $\boldsymbol{K}^{-1}$ matrix for a regular mesh with 240 nodes and 810 tetrahedra. This example was remeshed at a node and the difference matrix $\left(\boldsymbol{K}^{-1}\right)^{\prime}-\boldsymbol{K}^{-1}$ was found. The sum of squares of the elements of this matrix is called the total power of difference. Then, it was seen that $21 \%$ of this power was concentrated on the rows/columns of $P$ for this instance. When we also included the secondary nodes, this ratio reached to $96 \%$. Experimenting with similar matrices revealed that including only primary and secondary effects can mostly approximate the final matrix with an error less than $\approx 5 \%$ in terms of power of difference.

We implemented this with a modified version of the boundary condition change function which only calculates $L_{12}, L_{21}$, and $L_{22}$ of (2.29). In there, $L_{11}$ is the effect of other nodes' forces
$\left(\boldsymbol{F}_{1}\right)$ on other node's displacements $\left(\boldsymbol{U}_{1}\right)$. Neglecting $L_{11}$ in the boundary condition change reduces the complexity of each switch to $\mathcal{O}(m n)$. Consequently, this eases the mesh adaptation to $\mathcal{O}\left(m^{2} n\right)$. However, this method needs a better validation and verification. Extreme case with strange meshes (e.g., having slivers) should also be studied in depth.

### 2.7 Needle Model

In the simulation, the needle is represented by line segments which are connected to each other at the needle knots. In Figure 2.6, each $P_{i}^{N}$ is a needle knot and each $P_{i}^{N} P_{i+1}^{N}$ is a needle segment. As described in Section 3.5, each knot is defined as the controid of a needle section interface so that the sections are dual for segments in the FEA.

The needle flexibility is modeled using 3D FEM which was derived from [40]. A quadratic strain solution is applied on this model so that the relationship between the needle node displacements and forces are expressed as follows:

$$
\begin{align*}
& \underline{\hat{\mathbf{f}}}=\hat{\mathbf{K}}(\underline{\hat{\mathbf{u}}}) \underline{\hat{\mathbf{u}}} \\
& \Rightarrow \underline{\hat{\mathbf{f}}}=\left[\hat{\mathbf{K}}_{\ell}+\hat{\mathbf{K}}_{q}(\underline{\hat{\mathbf{u}}})\right] \underline{\hat{\mathbf{u}}} \\
& \Rightarrow \underline{\hat{\mathbf{f}}}=\hat{\mathbf{K}}_{\ell} \underline{\hat{\mathbf{u}}}+\underline{\hat{\mathbf{f}}}_{q}(\underline{\hat{\mathbf{u}}}) \tag{2.30}
\end{align*}
$$

where the subscript $\ell$ and $q$ denotes the linear and quadratic terms, respectively. The hat ( ${ }^{\wedge}$ ) is used to denote the needle vectors and matrices. For force boundary conditions, the equation above can be solved iteratively as shown by DiMaio [40] as follows:

$$
\begin{equation*}
\underline{\hat{\mathbf{u}}}_{i+1}=\hat{\mathbf{K}}_{\ell}^{-1}\left[\underline{\hat{\mathbf{f}}}-\underline{\hat{\mathbf{f}}}_{q}\left(\underline{\hat{\mathbf{u}}}_{i}\right)\right] \tag{2.31}
\end{equation*}
$$

Thus, if the forces on the needle model is known, its quadratic model displacements can be approximated iteratively. Next section shows how those needle mesh forces are found. However, since both needle and tissue are modeled using quasi-static elastic material, their coupling obviously needs to be damped for a stable behaviour. The maximum needle base manipulation speed and any internal step input such as the force during the node repositioning are the two dominant factors determining the minimum damping value to be used. The needle damping values are found empirically and applied on the needle forces of the next section before putting in (2.31).

### 2.8 Projecting Forces onto the Needle

The working nodes are the only points at which needle constraints are applied onto the tissue. Moreover, the FEA of the tissue solves for only those nodes' forces which are then applied back onto the needle to achieve needle-tissue coupling. However, similarly the needle model can accept constraints only at its mesh nodes, which are represented by the needle knots in the segmented
approximation. Thus, while the displacements of the contact nodes are derived from the needle knots, the reaction forces of the tissue mesh are also to be projected onto these knots.

Figure 2.6 shows an example with two working nodes $P_{1}^{W}$ and $P_{2}^{W}$ on the needle shaft. They can be either stuck or sliding on the needle segments $P_{1}^{N} P_{2}^{N}$ and $P_{2}^{N} P_{3}^{N}$. Assume that the forces caused by tissue deformation on those working nodes are solved by FEM and found to be $F_{1}$ and $F_{2}$. Then, the force on each working node is linearly distributed onto two closest knots as:

$$
\begin{align*}
& F_{1}^{\prime}=\frac{F_{1} \cdot\left|P_{1}^{W} P_{2}^{N}\right|}{\left|P_{1}^{N} P_{2}^{N}\right|} \quad F_{1}^{\prime \prime}=\frac{F_{1} \cdot\left|P_{1}^{W} P_{1}^{N}\right|}{\left|P_{1}^{N} P_{2}^{N}\right|} \\
& F_{2}^{\prime}=\frac{F_{2} \cdot\left|P_{2}^{W} P_{3}^{N}\right|}{\left|P_{2}^{N} P_{3}^{N}\right|} \quad F_{2}^{\prime \prime}=\frac{F_{2} \cdot\left|P_{2}^{W} P_{2}^{N}\right|}{\left|P_{2}^{N} P_{3}^{N}\right|} \tag{2.32}
\end{align*}
$$

The forces projected on the needle knots then have to be summed up (e.g., force on $P_{2}^{N}$ is $F_{1}^{\prime \prime}+F_{2}^{\prime}$ ).


Figure 2.6: Projecting the working node forces onto the needle knots.

### 2.9 Integrating Needle Forces

For a haptic implementation of the simulation, the force feedback at the needle base has to be calculated in order to be applied on the user's hand. This feedback is equal to the sum of the forces along the needle, in addition to their wrench effects as seen in Figure 2.7. Since it is very unlikely to have an axial rotation torque around the shaft, $\tau^{y}$ is neglected.

The forces can easily be added up for all $\hat{s}$ needle segments to find the total base force as follows:

$$
\begin{equation*}
F^{x}=\sum_{i=1}^{\hat{s}} F_{i}^{x} \quad F^{y}=\sum_{i=1}^{\hat{s}} F_{i}^{y} \quad F^{z}=\sum_{i=1}^{\hat{s}} F_{i}^{z} \tag{2.33}
\end{equation*}
$$

Calculation of the torque needs the distance from the base, which can easily be found in terms of the knot position in the local needle frame. Thus, the total torque becomes:

$$
\begin{equation*}
\tau^{x}=\sum_{i=1}^{\hat{s}}-F_{i}^{z} \ell_{i}^{y} \quad \tau^{z}=\sum_{i=1}^{\hat{s}} F_{i}^{x} \ell_{i}^{y} \tag{2.34}
\end{equation*}
$$



Figure 2.7: Integration of knot forces and torques on the base.

### 2.10 Discussion

The bottleneck of the presented approach is the time complexity of the mesh adaptation algorithm. Its dependency on node valence is also a concern, which constrains the mesh that can be used. Section 5.7 discusses implementation solutions regarding the speed of remeshing. Those proposed approaches also address the issue of current visualization calculations inside the haptic loop.

Instead of mesh adaptation, node addition techniques can also be considered. For instance, in Figure 2.4, $P^{\prime}$ could be added as a new node to the mesh. This would require the removal of two elements $\left(P P_{1} P_{2}, P P_{2} P_{3}\right)$ and addition of four new ones $\left(P P_{1} P^{\prime}, P_{1} P_{2} P^{\prime}, P_{2} P_{3} P^{\prime}, P_{3} P P^{\prime}\right)$. Although in 3 D six elements are needed instead of four, this method at least ensures a constant running time regardless of node valence as opposed to the current adaptation technique. Using this proposed addition with a Woodbury formula update could also make use of a hard-coded algebraic matrix inversion of rank 18 ( 6 involved nodes in 3D).

Finally, the issue related to the undeterministic nominal position of the desired new node $\left(P^{\prime}\right)$ can be dealt with force interpolation techniques over iteration cycles so that the user will not feel the discontinuity. DiMaio [40] reported these as force spikes without discussing their causes. Nevertheless, they have not resulted in poor haptic performance even without employing any remeshing (just snapping the closest node onto the needle). Although these spikes rendered some problems with our 3D flexible needle, they were overcome by damping (Chapter 5).

The iterative solution of the needle in (2.31) has some stability issues for certain ranges of Young's modulus and Poisson's ratio. This may need to be addressed using different solution techniques or some approximations that can be validated using experimental needle deflections. Some preliminary work which is not presented in here has been done on this issue. This work has shown considering other needle bending models could be a promising (fast and accurate enough) approximation. This has to be extensively addressed in a future work.

## Chapter 3

## Mesh Generation

The performance of FEM depends on the element discretization geometry which is called the mesh. In [58] Fleischmann et al. state that "for the finite element method, geometrical quality criteria suffice for most applications". There are many techniques and applications for creating those meshes. However, for a (fast) haptic implementation also a small number of nodes is desirable. This chapter shows how traditional Delaunay tessellation may perform poorly in those aspects due to our irregularly-spaced relatively-dense prostate surface definition. As a result, an advancing fronts implementation which also enabled us to set element sizes for different regions is described later in the chapter.

The above mentioned geometrical quality criteria can be defined in various ways. Usually aspect ratio or dihedral angle are considered. They are discussed in Section 3.3 on our final generated mesh. Throughout this chapter the tackled problem is converting the contour polygons to a surface definition and generating a quality mesh with small number of nodes which are concentrated around our region of interest (being the common needle insertion paths) with a conforming boundary on this surface.

After presenting our results local element renumbering which is needed as a final tissue meshing step for our implementation is described briefly. Finally our needle mesh is introduced.

### 3.1 Prostate Surface Construction

The contour data of the prostate slices are stored in the database of the BCCA. For an insight to the characteristics of the available contour data the treatment planning is first described briefly. This data was to be obtained for further processing. Then using these points which are lying on parallel slices a prostate surface definition was to be generated. Knowing this prostate boundary the entire region could be meshed accordingly.

### 3.1.1 Treatment Planning

The brachytherapy operation planning starts with a TRUS volume study of the prostate weeks prior to the procedure. This ultrasound examination is recorded on VCR. 10 to 12 parallel prostate slices are captured from this VCR and saved as 640x480 images in the BCCA patient database. During the following couple weeks, prostate segmentation, procedure planing, and its approval phases are carried out on a radiation therapy planning software called VariSeed ${ }^{\mathrm{TM}}$. First a doctor segments the prostate on each captured ultrasound image by identifying its contour. During this segmentation the pixels of the image are clicked using the mouse so that these clicked pixels will become the corners of a closed contour polygon. Since the prostate boundary is not clear especially around its apex, the physician also relies on the anatomical knowledge. In the second phase of planning, different radiation therapists prepare implantation plan proposals for the same patient. Then the physician that will perform the procedure chooses one of them considering dosimetry and the cancer level and location.

### 3.1.2 Contour Extraction via DICOM-RT

The easiest way of extracting data from the database was exporting in DICOM-RT (Digital Imaging and Communications in Medicine - Radiation Therapy) format in VariSeed. This format stores the images with their relative spatial information. Two additional files contain contour and dosimetry data. The contour file was read by the dicomread function of Matlab and a script was written to extract the contour corner points of the requested anatomy into an array. The anatomical structures segmented in the TRUS volume studies (of 20 study cases mentioned in next chapter) are the urethra, the prostate, and the rectum. Dicom contour file stores contour polygons (sets), volume image slices, and the anatomical structures in separate object structures since not all the images contain a contour polygon and not all the anatomical structures of interest span exactly the same slices. This information is parsed and rearranged using the Dicom manual from NEMA (National Electrical Manufacturers Association).

### 3.1.3 Culling the Contour Nodes

The resolution of ultrasound images is determined as 55.3 pixels/cm using the grid in the image, where the spacing is known to be 0.5 cm . When the contours are drawn over these images, it is seen that there are very close nodes $(<0.5 \mathrm{~mm})$ probably caused by the physician clicking the mouse button twice. There is also a node abundance such that many nodes are located almost linearly. This superfluity of nodes remains in the final mesh, since the meshing algorithms often add nodes on top of the current ones. Therefore, it was decided that some of the nodes could be eliminated initially without the loss of much anatomical information.

Considering the prostate is a smooth structure having ellipsoid-like cross-sections, very closely spaced nodes are not carrying much spatial information. A Matlab code was written to iteratively remove the contour nodes, which are closer to the rest of the nodes than a set threshold. When
two nodes are close, the one which would leave less gap after its removal is chosen to be removed. For example, if the minimum gap is between the nodes $P_{i}$ and $P_{i+1}$ at an iteration of this algorithm, then the distances $\left|P_{i-1} P_{i}\right|$ and $\left|P_{i+1} P_{i+2}\right|$ are compared and depending on which is smaller, either $P_{i}$ or $P_{i+1}$ is removed from the node list of that slice. So, for the given example in Figure 3.1 since $d_{i-1}<d_{i+1}, P_{i}$ will be removed (otherwise $P_{i+1}$ would be removed) leaving the list of remaining nodes as $\left\{\ldots, P_{i-2}, P_{i-1}, P_{i+1}, P_{i+2}, \ldots\right\}$. This is applied iteratively until there is no more nodes spaced less than the set threshold on a slice. The same procedure is repeated for each slice separately.


Figure 3.1: Node culling example.

A sample case having 262 nodes is downsized to 194 nodes using a threshold of 3 mm , to 160 with 4 mm , and to 142 nodes with 5 mm . Figure $3.2(\mathrm{a})$ shows a slice where the circles indicate the removed nodes and the stars are the culled ones after the application of our algorithm with 4 mm threshold. For the same case the nodes of all slices are shown in Figure 3.2(b). Note that most of the closely spaced nodes are around the base and the apex. This is due to the physician inputting almost as many contour nodes for those regions as the other slices, although the contours have shorter perimeter here. Also these figures show the nodes which are probably erroneously added due to the physician double-clicking or clicking twice (instead of single-click). As seen those are also removed by this algorithm (an example is at the coordinate [50,30] in Figure 3.2(a)).

A 5 mm culling threshold was chosen. One reason was the inter-planar distance being 5 mm which sets our minimum sampling in one dimension. The other was the qualitative observation of the final mesh. Although the number of nodes increased significantly with decreasing threshold, the surface and the mesh did not change much visually.

### 3.1.4 Obtaining the Surface Definition

The culled nodes are points lying on parallel slices. A first attempt was direct meshing from those surface points. However, simple Delaunay tessellation of all these nodes in Matlab gave us tetrahedra outside this region since the prostate surface was apparently not convex. Although there are algorithms to remove the elements outside an object, for any algorithm to work on this data first the surface definition was needed in order to describe the connectivity of those nodes

(a) node culling on slice "Base +1.0 "

(b) All surface nodes on 11 contours

Figure 3.2: Node culling with a threshold of 4 mm on a sample data. Circles are the removed nodes and the stars are the remaining ones.
on the surface. Thus, simple tessellation did not work and the problem is divided in two parts, surface generation and volume meshing. This section presents the solution to the first part.

We attempted to surface construction by tessellating the nodes of each two consecutive planes separate from the others and then we were to accumulate the triangles that would be on the prostate surface. Although identifying those triangles was theoretically possible, it did not have a straight-forward solution either. Thus, the solution to this was searched in the literature.

Keppel [91] has one of the earliest publications on surface reconstruction from planar slices. For each two neighbour nodes on a slice, he looked for a third one in the adjacent slice and formed the surface triangles this way. Various researchers have proposed methods dealing with contour tiling and branching problems, where the former refers to finding better-conditioned surface triangles and the latter addresses the differences in number of contour polygons on adjacent slices. Boissonat [92] proposed a method based on the Delaunay tessellation of each pair of slices. The method then extracts the boundaries of each slab, once their volumes are tetrahedralized. This is very similar to what we attempted above and its code (Nuages ${ }^{\text {© }}$ [93] by INRIA) is available on the internet.

Nuages gets a set of simple closed polygons on parallel cross-sectional planes and outputs the set of surface triangles or volume tetrahedra representing the 3D polyhedra. The tetrahedral elements generated by simple Delaunay methods are generally poor-conditioned for FEM. Moreover, this software cannot generate CDTs and earlier trials of meshing the two regions separately faced the problem of non-conforming surface triangulation of the tissue tessellation inside and outside the prostate. Therefore, only the surface output is obtained using this software and this triangulated surface is meshed in a further step.

The culled nodes are written in a file format that Nuages can read and a $d x f$ file is requested as the output format for the triangular surface representation of the object, since this native

AutoCad ${ }^{\circledR}$ file format can be imported by many popular meshing software. For the sample case mentioned above with 142 culled nodes, a surface definition is generated consisting of 280 triangles as shown in Figure 3.3.


Figure 3.3: Nuages surface output ( 280 triangles) for 140 input nodes on 11 parallel slices.

### 3.2 Meshing the Region

First, element sizing was to be determined. Many meshing studies employ larger elements inside the objects based on Nicolson's empirical observation from his simulations in [94] that the spatial cut-off frequency of an elastically deformable body in response to external loads decreases faster than $1 / d$ in terms of the depth $d$ from the surface of the body. This result suggests that element sizes that are proportional to the depth can be accurate enough for many surface interaction simulations such as in [33]. However, since our model is manipulated via inner nodes, the same statement does not necessarily apply to our case. That is why a low-resolution mesh with similar element sizes regardless of the depth was aimed during the mesh generation for our simulation.

Various software packages were considered for meshing the surface definition. Octree-based ones were neglected at the first place since they would have small elements on the prostate boundary. Many programs which cannot generate a boundary constrained mesh are also neglected since at least the prostate needed a conforming boundary separating it from the surrounding tissue.

A good and easy to use option seemed the TetGen [95] package since it offers CDTs with high-quality tetrahedra for FEM. The C++ source code compiled on Linux accepts a polyhedral definition of the object and can output the mesh in various popular file formats. The mesh can be constrained on the minimum radius-edge ratio and the maximum element volume, and a post-processing option can attempt to remove slivers with high dihedral angles.

The surface definition ( $d x f$ file) is translated into TetGen input format by a Matlab script. When its output is visualized, it is seen that with any quality setting many tiny tetrahedra on the prostate surface are generated. Moreover, the non-optimal surface triangulation generated by

Nuages is almost not touched. It is concluded that this package is intended for small deformation solid modeling, since its underlying technique aims for feature preservation and gradual element sizing outwards to the surface. Thus, it ends up with $\approx 3000$ nodes just for the interior of the prostate. In addition, the surrounding tissue requires mesh nodes as well. This made us aware that not many packages are intended to mesh for low number of elements.

Owing to its extensive FEM analysis options, ANSYS ${ }^{\circledR}$ was as a strong candidate which was tried next. However, its complicated quality settings, relatively large number of generated nodes, and individual installation and stability problems (experienced with version 7.0) led us to consider other solutions.

At this point, a package that was tried was found to satisfy the shortcomings of the above mentioned ones. GiD ${ }^{1}$ from CIMNE (International Center for Numerical Methods in Engineering) in Spain is a geometrical modeling, meshing, and visualization software. It uses advancing fronts so that a preferred element size setting can define the rate that the front advances (which in turn determines an average element height). It can also generate conforming boundaries allowing separate preferred element size settings for different material subregions.

The prostate was imported from the $d x f$ file. It was enclosed in a rectangular prism and different materials were attributed to the interior and exterior of the prostate. The width and the height of this rectangular prism were set to the dimensions of the ultrasound. A larger depth than the span of the volume study is meshed in order to account for the offset of the prostate underneath the skin. Considering the 5 mm culling threshold and slice separation, after experimenting with various values while observing the size of the resultant mesh, the tetrahedron edge lengths are chosen 7 mm for the prostate and 14 mm for the surrounding soft tissue.

The two ends of the prostate, the base and the apex, have relatively fewer surface nodes since the large polygon segmentations on these surfaces directly passes from Nuages and only one or two nodes are added by GiD. During the further steps, when this mesh is used for the needle insertion simulation, it is concluded that more nodes on these surfaces would create a better path for the needle. Since it especially passes through the apex quite frequently (for almost every insertion), having a finer mesh on it would improve the simulation. Therefore, a further step is added before volume construction, where the first and the last contours are displayed and the user may add extra surface nodes on them. For the specific example presented, 5 and 4 nodes are added on the base and the apex, respectively. This increased the total number of nodes from 152 to 161.

### 3.3 Results

The final generated mesh has 570 nodes and 2801 tetrahedra. Figure 3.4(a) shows the bounding box surface and the prostate surface triangulations. A rendered prostate image with the 4 added vertices at the apex can be seen in Figure 3.4(b). In this mesh, the smallest element volume

[^5]was $6.67 \mathrm{~mm}^{3}$, where the largest one is $719.41 \mathrm{~mm}^{3}$. The shortest and the longest edges were 3.06 mm and 24.04 mm , respectively. Cumulative histograms of these quality measures for all the mesh elements are presented in Figure 3.5. In those graphs the height difference of a bar from its left adjoining bar indicates how many elements that value applies. Thus, the higher the gradient is, the more elements are concentrated in that range of values. For instance, Figure 3.5(a) has a steep ascent from $\approx 300$ to $\approx 1600$ between 4.9 and 6.5 and almost flat above 14 , which consequently mean there are $\approx 1300$ elemens having a minimum edge about $5-6 \mathrm{~mm}$ and only a couple elements with an edge longer than 14 mm .


Figure 3.4: Mesh visualization.
Dihedral angle is the angle between two planes. For an element, minimum and maximum dihedral angles give a general idea of its flatness. A too small minimum or a too large maximum would indicate the presence of slivers. In our mesh, the smallest minimum is $9.94^{\circ}$ where the largest maximum is $159.49^{\circ}$. In TetGen meshing package, a sliver is defined for a maximum dihedral angle over $175^{\circ}$. Our maximum value is well below this quality limit. Moreover, the low rate of change for values higher than $135^{\circ}$ in Figure 3.5(b) and less than $19^{\circ}$ in Figure 3.5(d) show that there are very few low-quality elements.

Well-conditioned elements are very important in FEA, since the formulas and calculations supply better approximations for elements with geometries closer to an optimal shape. For tetrahedra, this shape is a regular tetrahedron which has four equilateral triangular faces. The performance of FEM degrades as tetrahedra deviate from this optimal structure. There are various criteria to measure this deviation. Aspect ratio is the common term used for this assessment factor, however it is definition changes at different resources. Generally it is a ratio of the circumsphere radius or the longest edge length to the inscribed sphere radius or the shortest edge length. We use the longest edge length divided by the diameter of the inscribed sphere. Table 3.1(a) presents the number of elements with certain aspect ratios. Our mesh having no


Figure 3.5: Mesh quality analysis.
elements with an aspect value larger than 15 is considerably good. Another element criterion is the radius-edge ratio which is the circumsphere radius divided by the shortest edge length. It is a weaker quality measure since it cannot identify many slivers. The values below 2 are preferable and our mesh has only 20 elements out of this limit as seen in Table 3.1(b).

### 3.4 Local Element Numbering

Some parts of the code requires a special local numbering of the element corners in order to make faster run-time operations. This corner ordering would make the decision of whether a point is inside an element or not. This basically needs to be checked for tip penetration in Section 2.5 and during ultrasound synthesis in Section 4.4. It also saves us from taking absolute values of the determinant, where the volume of an element is calculated using:

$$
v^{e}=\frac{1}{3!}\left|\begin{array}{llll}
x_{1} & y_{1} & z_{1} & 1  \tag{3.1}\\
x_{2} & y_{2} & z_{2} & 1 \\
x_{3} & y_{3} & z_{3} & 1 \\
x_{4} & y_{4} & z_{4} & 1
\end{array}\right|
$$

(a) Aspect ratio

| Aspect ratio | elements |
| :---: | ---: |
| $1.1547-2$ | 0 |
| $2-2.5$ | 31 |
| $2.5-3$ | 331 |
| $3-4$ | 1284 |
| $4-6$ | 989 |
| $6-10$ | 156 |
| $10-15$ | 10 |
| $15-\infty$ | 0 |

(b) Radius-edge ratio

| Radius-edge | elements |
| :---: | ---: |
| $0-0.707$ | 253 |
| $0.707-1$ | 1566 |
| $1-1.1$ | 298 |
| $1.1-1.2$ | 222 |
| $1.2-1.4$ | 240 |
| $1.4-1.6$ | 113 |
| $1.6-1.8$ | 67 |
| $1.8-2$ | 22 |
| $2-2.5$ | 14 |
| $2.5-3$ | 1 |
| $3-10$ | 5 |
| $10-\infty$ | 0 |

Table 3.1: Mesh quality.

If the corners are in right-hand ordering, (3.1) gives a positive volume. Right-handed local node numbering is geometrically equal to having the node $P_{4}^{e}$ on the same side with the surface normal, $\overrightarrow{P_{1}^{e} P_{2}^{e}} \times \overrightarrow{P_{1}^{e} P_{3}^{e}}$, of the plane passing through $P_{1}^{e}, P_{2}^{e}$, and $P_{3}^{e}$ as seen in Figure 3.6. So, the local numbering of element corners generated by GiD is processed by a Matlab script to ensure right-handedness before simulation starts. For each element, first its volume is calculated using (3.1). If it is positive, the order is not touched. Otherwise, the nodes $P_{3}^{e}$ and $P_{4}^{e}$ are switched, which puts the element in right-handed form, since it changes the side of $P_{4}^{e}$ with respect to $P_{1}^{e} P_{2}^{e} P_{3}^{e}$ plane.


Figure 3.6: A right-handed tetrahedron.

### 3.5 Mesh for the Needle Model

The needles used in prostate brachytherapy procedures are long and very flexible. For instance, Figure 3.7 shows the last 16 cm of a needle straight and bent. For a 0.6 N orthogonal force the lateral deflection is more than 6 cm while the deformation is still elastic. Their flexibility complicates the insertion making the prediction of the tip motion harder. Nonetheless, the flexibility is a desirable property that also makes the needles steerable to reach occluded targets. This steering challenge is incorporated into our simulator by modeling the needle flexibility.


Figure 3.7: Brachytherapy needle deflection under a lateral tip force.
Different models including beam elements, an analytical cantilever beam model, and spring models have been considered. Because of its physical validity and previous successful implementation as a needle model [40], finally FEM was chosen. The needle mesh is prepared manually using the same type of (TET4) elements as the tissue model. This enables us to use the same basis functions, solution techniques, and programming procedures for the needle and the tissue.


Figure 3.8: Needle mesh topology.
The criteria we chose as the guidelines for the needle mesh construction were filling the needle volume using the least possible number of flexibility-representative elements, in order to decrease computational time, and keeping at least one surface between adjacent elements, in order to rotationally fix them. A simple uniform structure which comes to mind at first is shown in Figure 3.8. However, experimenting with this implementation revealed how susceptible FEA is to the axial asymmetry of these elements and their connectivity. Especially with few elements,
the ratio of tip deflection caused by equal-magnitude forces in perpendicular axes raises up to 2:3. That means a force applied in pure +Z direction causes an asymmetric tip deflection of 2 units in -X , in addition to the expected 3 units in +Z . To avoid this a new structure was designed in which the elements are connected in a way to cancel out each other's asymmetric effect.

Figure 3.10 shows the tip deflection due to a constant lateral force rotating around the tip for both models. Figure 3.9 presents sections of both mesh designs. Throughout this thesis the term section denotes a building block of needle mesh so that similar blocks can be stitched at the end of each other to form the entire needle shaft. Needle knots are virtual points (for interaction model) lying at the interfaces of these mesh sections. Thus, each needle segment, which connect consecutive needle knots, represent a line passing through the center a needle section.


Figure 3.9: Mesh topology of a section for two different designs.


Figure 3.10: Lateral tip deflection with respect to number of sections.

### 3.6 Discussion

A method for generating a volume mesh from contours on parallel slices has been presented. Considering many seeds are implanted in the prostate, a finer mesh is preferred in it. Although only the prostate is defined for fine meshing in the current process, the entire PTV and the soft tissue in front of it can also be meshed fine considering the frequent needle interaction.

The values for node culling threshold and preferred element size were found empirically. To achieve a small number of nodes for high speed in haptics, various applications and techniques were attempted. The final prostate mesh generated which has a reasonable number of nodes was shown to have good-quality elements for FEA.

Both the FEA in Chapter 2 and the mesh generation scheme presented in this chapter are applicable with more than two tissue regions. Thus, the segmentations from different medical imaging modalities can also be used to generate the meshes for other anatomical structures inside the volume of interest. These meshes can be incorporated into the simulator. This could possibly bring issues especially for the bladder and the pubic arch. The bladder can be modeled as a fluid pocket [42]. Generally, the bones can generate problems because they have relatively high stiffness and represent fundamental boundary conditions. The high stiffness results in large force discontinuities when a node is snapped (see Section 2.6). Therefore, the pubic arch may require special treatment (such as solid contact models) out of the FEA.

The presented FEM asymmetry problem on the needle mesh was overcome by a new design. It was shown to be symmetrically modeling the bending of a cantilever structure using a small number of optimally structured elements throughout its volume. This finding also made us aware that a simple uniform tetrahedralization of the tissue could result in asymmetric behaviour. That was why uniform meshes were not considered as the first tissue meshing solution.

## Chapter 4

## Ultrasound Synthesis

According to the instantaneous deformed layout of the tissue, TRUS images are to be generated and displayed to the user. This requires proper deformation and interpolation models for fast realistic image synthesis, accounting for the characteristics of ultrasound (e.g., speckle). Seeing the needle reflection on these synthesized images is also important for the physician in judging where the needle path lies. Therefore, in the following sections we propose a TRUS slicing method with an integrated needle reflection model. A typical brachytherapy procedure was summarized in Section 1.2. Below the BCCA study during which our dataset was acquired is described.

### 4.1 Patient Data Collection

For a research study at the BC Cancer Agency, 20 prostate brachytherapy cases were to be scanned in CT and MRI right after each procedure. The data collected from these cases would be beneficial for us in segmenting all the tissue types around the prostate. These regions can be incorporated into our simulator. Moreover, the segmented seed locations from these cases could be used to validate the designed simulator. Grabbing the ultrasound data for the same cases would also enable us to validate other aspects of our work. For example, the seed and the needle shaft overlays onto TRUS could be evaluated. The TRUS image of the deformed region during an insertion could also be compared with the synthesized B-modes. Thus, we captured intra-operative ultrasound data using high resolution video hardware. In order to know the instantaneous stepper position (consequently, the slice depth) of a TRUS image and the instantaneous motion of the needle base which are needed for the above mentioned validations, we recorded a frame including the precision stepper, the needle, and the physician's hand holding onto it using a digital camcorder. This data could also help us verify needle flexibility and deflection models. Furthermore, tissue elasticity parameterization (elastography) and/or verification of its known biomechanical values can be performed on this captured data.

The study conducted in the BCCA aimed to compare the dosimetry calculations using USCT (Computed Tomography) fusion with the ones obtained from MRI-CT (Magnetic Resonance

Imaging) fusion. Thus, unlike typical procedures for these 20 cases prostate ultrasound volume studies were conducted at the end of each procedure. They segmented the urethra in all modalities and registered it using VariSeed for both fusion types above. A catheter with contrast fluid is used to make urethra visible in TRUS images. By means of this registration, the seed locations easily found in CT (see Figure 4.1) can be projected on MRI where soft tissue can more easily be segmented (see Figure 4.2). We also recorded this volume study, which gave us TRUS slices scanned in 5 mm increments with all implanted seeds from the base to the apex.


Figure 4.1: CT images after a brachytherapy procedure.


Figure 4.2: MRI after a brachytherapy procedure.

### 4.2 The Shift and the Torque

The forces applied by the needle cause the tissue to displace and deform. This is mainly observed during the insertion when the forces on the tissue are higher because of the tip cutting force added on the shaft friction. Although the insertion and the retraction mainly deforms the region parallel
to the needle axis, the mesh geometry and the bonding between anatomical structures may also cause more complex behaviour (e.g., sagittal torquing), details of which are discussed later in this section.

Although a sagittal imaging option is available in the TRUS hardware, the images are mostly acquired transversally to observe the prostate cross-sections while the stepper that positions the probe is moved with 5 mm increments. The instantaneous ultrasound image represents tissue content residing momentarily in a slab. Meanwhile, when the tissue deforms during needle insertion, an anatomy in a separate region can possibly shift into this slab. This effect is easily observed especially at the base of the prostate, where the TRUS image before the needle insertion is mostly covered with the bladder and it has almost no prostate tissue in it (see Figure 4.3(a)). As the needle penetrates into the prostate or while it is advancing through, the prostate base which basically just shifts in 3D suddenly appears in the image (see Figure 4.3(b)). Figure 4.3 shows two images of the same depth with $\approx 1 \mathrm{~s}$ temporal separation. This means the stepper (consequently the probe) is not moved between these two images, so their difference comes from the shift of the tissue in this one second. In the first image, the physician is not applying any force on the needle. Then, when he pushes the needle, the prostate base can be easily seen in Figure 4.3(b) (which is actually the same slice but just the prostate shifts into it). Note that the needle tip deflects upwards with the torquing prostate. This torquing effect is described below.


Figure 4.3: Prostate shifting into the bladder during a needle insertion.

Another observable deformation phenomenon occurs when TRUS is viewing a slice near the middle or the base of the prostate. If the needle is pushed meanwhile, the observed prostate cross-section moves upwards up to 1 cm . This is again because of the torque, which the advancing needle generates around the stiffer tissue connecting the prostate to the pubic arch.

For simplicity, the problem can be studied in 2D sagittal direction without loss of generality. In 2D, the TRUS imaging planes look like lines as shown in the simplified sketch in Figure 4.4. The physician can adjust the stepper to move the TRUS back and forth to see different slices. The medical naming convention for these TRUS slice depths are "Base", "Base +0.5 ", "Base +1.0 ", etc. Let these lines (which are planes in 3D) be $L_{1}, L_{2}, \ldots, L_{n}$ before any deformation (needle insertion). Note that after deformation the TRUS probe will obviously still be acquiring planar
(linear in 2D) data whereas the tissue contents of the voxels on those planes will not be the same as $L_{i}$ anymore.


Figure 4.4: A 2D sketch of the region and the TRUS images before needle intervention.

When the needle is being inserted into the prostate, the structural deformation can be demonstrated as in Figure 4.5. With the friction and the cutting forces of the needle, the prostate, which is relatively stiffer than the surrounding tissue, shifts axially and rotates sagittally (due to the stiff tissue above its apex), such that $\Omega$ will shift and occupy $\Omega^{*}$ at a time instant.


Figure 4.5: Demonstration of the structural deformation during needle insertion.

The first observed phenomenon, the base shift into the bladder, can be easily visualized considering the $L_{1}$ in Figure 4.4: while we do not normally see the prostate in $L_{1}$, after the shift the base will be visible in it. Let us also analyze the second observed phenomenon, the upward image shift due to the torquing effect. Let $L$ be the slice imaged by the TRUS probe. Assume that before the needle insertion the pixel $P$ of $L$ is inside the bladder. Consequently, let us say that the physician sees $P$ almost black (due to the fluid filled bladder). While the needle is being inserted, the sagittal torquing may have the prostate occupy the coordinate of $P$. If this happens, $P$ will then show prostate tissue (since $P \in \Omega^{*}$ ) changing the TRUS image look. In
other words, the cross-section of $\Omega$ is then be shifted upwards showing the cross-section of $\Omega^{*}$.

### 4.3 Problem Definition

Due to the above mentioned deformations, the initially observed lines $L_{i}$ that denote the tissue content at those locations change their shape during the insertion. To visualize this better, a 3D sketch ${ }^{1}$ of the tissue with an advancing flexible needle is shown in Figure 4.6. Here a representation of how the TRUS planes may deform is given. We will refer to the deformed geometries of these TRUS slices as $L_{1}^{*}, L_{2}^{*}, \ldots, L_{n}^{*}$. In this chapter, the superscript "*" is used for the geometries in deformed mesh layouts.


Figure 4.6: A sketch representing the 3D deformation of the initial TRUS images.
Note that whereas $L_{i}^{*}$ can be curved because of the deformation, any instantaneous TRUS imaging plane $L$ will always be linear (and similarly planar in 3D) since the probe acquires planar images regardless of the tissue content.

The ultrasound data on slices $L_{i}$ with a 5 mm separation is acquired during the preoperative TRUS volume study. Thus, the voxel intensity values of the initial mesh is known on those planes $\left(L_{i}\right)$. Let us assume that "the voxel intensities of $L_{i}$ are representative for the corresponding voxels of $L_{i}^{*}$ in the deformed state". This is not always true, since US image pixels are in fact a cumulative effect of scattering and reflections. This refer to two aspects of the ultrasound imaging, the speckle formation and the imaging directionality. The former affects us in a way that the same voxel (tissue content) shifted to another part of the volume is not necessarily be denoted by the same image intensity on an US image. For example, comet tail which is observed behind most of the metallic surgical tools is formed along a line originating from the probe surface, therefore it is orientation dependent. Also acoustic shadowing of different density anatomical structure depends on the angle of US incidence.

As a result, the intensity of a point may not convey much information about the tissue. Nevertheless, prior work $[86,84]$ achieved success using interpolation for reslicing. However,

[^6]similar studies in literature have dense input datasets as opposed to our 5 mm inter-slice spacing.
In brief, the problem can be formalized as "determining the ultrasound image on $L$ at any instance of time with the ultrasound data given on the initial slices $L_{i}$ which are deformed into $L_{i}^{*}$ at that instance". Two approaches and their comparison are presented below. The reader is referred to Appendix A. 4 for information on basis functions and discretization which are mentioned in this chapter.

### 4.4 A Direct Approach

A simple instance of the problem is sketched in Figure 4.7, where the imaged plane $L$ falls completely in-between two adjacent deformed data slices. Thus, in order to interpolate the intensity on any point $A$ on $L$, the geometry of $L_{i}^{*}$ and $L_{i+1}^{*}$ are needed. The points of these slices can be found using Algorithm 4.1.


Figure 4.7: Demonstration of the structural deformation during needle insertion.

```
Algorithm 4.1 Finding the locations of the input data after deformation.
    for \(\boldsymbol{x} \leftarrow\) each voxel on \(L_{i}\) do
        Determine the mesh element \(\Omega^{e}\) where \(\boldsymbol{x} \in \Omega^{e}\)
        Find the basis functions \(N_{i}^{e}\) for the element \(\Omega^{e}\)
        Using the corner displacements \(\boldsymbol{u}_{i}^{e}\), find deformed position \(\boldsymbol{x}^{*}\)
```

This algorithm is suitable for various optimizations. For instance, on line 2 of Algorithm 4.1 the elements containing the data points are needed, and this can be calculated offline and put in a look-up table. In other words, in an offline step all input data points, which are on the planes $L_{i}$ with depths known prior to the simulation, can be tagged by the element numbers
they reside in. In addition to this, the basis functions already found in offline $K^{e}$ computations can be stored in arrays for fast access on line 3 of Algorithm 4.1.

Determination of whether a point is inside an element or not is performed using the function in Algorithm 4.2. An instance for one loop of this algorithm, where $X$ is in the element can be seen in Figure 4.8. Note that it requires the local numbering of the elements to be in right-hand order as described in Section 3.4.

```
Algorithm 4.2 The function for deciding whether a point is inside a tetrahedron.
    function IsPointInsideElement ( \(X, e\) )
        for all Face \(\in e\) do \(\quad \triangleright\) For each face of the element
            \(\left\{P_{l}, P_{m}, P_{n}\right\} \leftarrow\) corners of Face \(\triangleright P^{\prime}\) s ought to be in right-hand order
            \(\vec{N} \longleftarrow \overrightarrow{P_{l} P_{m}} \times \overrightarrow{P_{l} P_{n}} \quad \triangleright\) Find the surface normal
            if \(\left(\overrightarrow{P_{l} X} \cdot \vec{N}<0\right)\) then return False \(\quad \triangleright\) Dot product finds the projection
        end for
        return True
```



Figure 4.8: Determining which side of the plane $P_{l} P_{m} P_{n}$ the point $X$ is. This is one of the four checks before $X \in e$ conclusion.

After calculating the positions of the known intensity values in the deformed mesh using Algorithm 4.1, the intensities on $L$ can then be interpolated. This interpolation could be defined in numerous ways.

An inherent nature of ultrasound is the speckle. Its intensity and distribution inside a tissue region is caused by the superpositioning of the sound reflecting from unidentifiable scatterers in a complex manner. Therefore, these B-mode speckles (the dark/bright spots) do not really convey much information about the underlying tissue. Thus, their spatial interpolation from coarsely sampled data using most common techniques may just give too blurred images where black and white speckles gray out each other.

To properly interpolate the data at point $A$, the points on which the interpolation will work on
and their weightings on the output have to be determined. For such problems there are scattered data interpolation techniques available [77]. Pixel nearest neighbourhood (PNN), voxel nearest neighbourhood (VNN), radial basis functions (RBF) [80] are some of the methods that were attempted in literature.

A valid approach is finding the closest points, $B_{i}$ and $B_{i+1}$, on each curve (see Figure 4.7). However, this would bring an extreme computational burden. An approximation could be the use of points $C_{i}$ and $C_{i+1}$. They are the transversally (height-wise in 2D) closest points of $L_{i}^{*}$ and $L_{i+1}^{*}$ to $A$.

Once the points are determined, the intensity of $A$ can be found by linear or quadratic interpolation. For instance, linear interpolation can be applied as follows:

$$
\begin{equation*}
I_{A}=\frac{I_{B_{i}}\left|B_{i+1} A\right|+I_{B_{i+1}}\left|B_{i} A\right|}{\left|B_{i} A\right|+\left|B_{i+1} A\right|} \tag{4.1}
\end{equation*}
$$

where $I_{P}$ denotes the intensity of $P$.
One drawback of this direct approach is the algebraic complexity in finding the proper points (ideally the closest ones, e.g., $B_{i}$ and $B_{i+1}$ for $A$ ) on the neighbouring curves. Further disadvantages arise when $L$ cuts one or more of $L_{i}^{*}$, which happens if the tissue shifts more than 5 mm . In this case, more than two $L_{i}^{*}$ need to be found by Algorithm 4.1. Dealing with all separate cases complicates the problem algorithmically. Therefore, the following simpler yet accurate approach was followed.

### 4.5 Inverting the Problem

This ultrasound synthesis problem can be tackled by inverting the steps above, so that the points to be interpolated from would be in a grid structure letting us use the regular-grid interpolation techniques which are fast and well-known. Considering the input TRUS slices already have regularly-sampled voxels on them, leaving those as they are might be beneficial. Thus, let us imagine the deformation of the tissue as a backward deformation of the imaging plane $L$ as seen in Figure 4.9. This is equal to finding the set of initial (nominal) mesh points, which the given deformation would relocate onto the imaging plane. Note that "a surface that is fit on those points should look similar to the mirror image of the deformed ultrasound input data planes".

The positions of the imaging plane points can be transformed from the deformed mesh to the nominal mesh using a similar algorithm to Algorithm 4.1. However, this time the basis functions have to be calculated at every run, meaning every iteration. That is simply because the transformation is now from the deformed elements to the nominal ones and the basis functions of these deformed elements change with their changing geometry as the tissue is manipulated by the needle. Figure 4.10 demonstrates an instance, where $P_{i}^{e}$ are the nominal corner positions of the element $e$. The dotted lines denote the nominal mesh whereas the solid lines depict the deformed geometry with vertices $P_{i}^{e^{*}}$.


Figure 4.9: Ultrasound imaging plane transformed into the nominal mesh.

Algorithm 4.3 presents the pseudo-code for this deformed-to-nominal transformation. For each point $X^{*}$ on the imaging plane $L$, it first finds the deformed element $e^{*}$ that confines $X^{*}$ (lines 2-3). Then using the basis functions of this element $e^{*}$ and the negated displacements, $-\boldsymbol{u}_{i}^{e}$, the corresponding nominal position $X$ is calculated. In our implementation, each calculated basis function was stored in an array to speed up further references to the same element.

```
Algorithm 4.3 Finding the nominal positions of the imaging plane pixels.
    for all \(X^{*} \in L\) do
        for \(e^{*} \leftarrow 1\) to \(\# e\) do
            if IsPointInsideElement \(\left(X^{*}, e^{*}\right)=\) True then break
        Find basis functions \(N_{i}^{e^{*}}\) for the deformed element \(e^{*} \quad \triangleright\) using \(P_{i}^{e^{*}}\) as \(\boldsymbol{x}\) in (A.18)
        \(X \leftarrow X^{*}+\sum_{i=1}^{4} N_{i}^{e^{*}}(\boldsymbol{x})\left(-\boldsymbol{u}_{i}^{e}\right) \quad\) Find nominal position for US data point using (A.17)
    end for
```

A drawback of this scheme is the low speed of the IsPointInsideElement function, which bottlenecks the current synthesis process. Although it was implemented in Matlab, even a C implementation with optimizations would be slow since checking all mesh elements is not an efficient approach. First a simple yet effective addition was made before Line 2 to limit this search only to the elements cutting the imaging plane. Elements having nodes on both sides of the imaging plane were found in $\mathcal{O}(n)$ computations. For example, in our current mesh (given in Chapter 3) the imaging plane passing through "Base+2.0" cuts 337 of the total 2801 elements and this can be found in relatively negligible time. This technique is open to many other further optimizations, some of which are:

- For each voxel, first the element of a neighbouring voxel (the parent element of which has already been found) can be checked considering that $X^{*}$ is most probably inside the same element with its neighbours
- Before traversing the elements, sorting the elements with respect to their proximity of $X^{*}$


Figure 4.10: An instance showing the imaging plane, the deformed mesh (solid lines), and its nominal geometry (dotted lines).
to their centroids increase the chance and speed of finding the right element. (This sorting could just be done once in a while)

- Element connectivity graphs can be used to make an educated guess of the next element trial.

To avoid element traversal, analytical methods could be employed as well. For example, finding the intersections of the element edges with the imaging plane gives the triangular patches of $L$ spanned by each element. We can then easily tag the points inside these patches.

Real-time consecutive image generation brings even more options of optimization, with the main assumption of relatively slow deforming tissue with respect to the iteration rate. TRUS is working at 12 Hz imaging rate during the operation. Although the current implementation cannot achieve this speed, it is anticipated that a C code using a combination of above mentioned methods can easily reach this rate.

### 4.6 Image Interpolation Results

For the inverted problem, the ultrasound was synthesized using the fast interp3 function of Matlab. The amount of input data is not noticeably changing the speed of this function. Therefore, instead of only the adjacent planes, the voxel locations and intensity values of all the initial data
planes were sent to the function input. As the output, the intensity values at the transformed positions $X$ are asked for.

For instance, in a deformed mesh an ultrasound synthesis at "Base+2.0" is an interpolation probably between the initial images of "Base+2.0" and "Base+2.5", considering the shift of the latter towards the base during needle insertion. Those initial images of a patient can be seen in Figure 4.11(a) and 4.11(b), respectively.

To see the interpolation effects caused by the deformation, the simulation implementation in Chapter 5 was used. For the given pubic arch-prostate bonding, an insertion at the lower left corner of the prostate twists the region coronally as well as sagittally. This caused half of the ultrasound image to be close to one input data plane and the other half to be closer to the other, which generated a good case to see interpolation artifacts.

First the nearest-neighbourhood interpolation was tried. In Figure 4.11(e) the fracture line separating the regions can easily be seen around the column $E$. This input plane switch artifact and other discontinuities caused by skipped image lines completely disappeared when linear interpolation was used as seen in Figure 4.11(f). Furthermore, the blurring caused by the interpolation was minimized using cubic interpolation as demonstrated in Figure 4.11(d). On the average interp3 finishes the linear interpolation in 0.6 sec and the cubic interpolation in 3 sec .

For all three interpolation schemes, it can easily be observed that the left side of the images bear features from "Base +2.0 " and on the right of the column $E$ they represent "Base +2.5 " more. To give examples on Figure 4.11, the hospital naming convention will be used. It names the row $1,1.5, \ldots, 6,6.5$, and the columns $A, a, \ldots, f, G$. So, let us say that the white spot around $e 6$ and the seed at $F 1$ on "Base +2.5 " in contrast do not exist on "Base +2.0 ". As expected, they are clearly visible in the generated images where this region of the imaging plane is still close enough to "Base +2.5 ". Similarly, the seed at a2 on the plane "Base +2.0 ", which is one of its distinguishable features, is observed on the interpolated image, whereas the seed of "Base+2.5" at $C 5$ does not show up at all as expected.

### 4.7 Needle Overlay

Intra-operatively, the main use of the ultrasound is seeing the placement of the needle and the seeds, rather than observing the deformation of the underlying tissue. That is why the needle appearance is also needed on the ultrasound images. As the seeds are implanted, they are also have to be added on the images. However, to show the concept of overlaying these info, only the needle shaft case is presented here and a discussion of seeds and the needle tip is included at the end. Since during the insertions, the needle shaft is mostly parallel to the probe, the rest of this section assumes that they are always aligned.

In the ultrasound image, the needle shaft looks basically as a bean-like bright spot followed by its comet tail. This bean-shape is caused by the orbital elongation of features under TRUS and it can be approximated with an ellipsoid curved around the probe. From here on, its slight

(c) needle overlayed on cubic

(e) nearest-neighbour

(d) cubic

(f) linear

Figure 4.11: TRUS slices (a,b), interpolation results (e,f,d), and needle overlay (c).
curvature will be neglected and it will be referred as the ellipsoid, which is a valid assumption especially distant from the probe. The comet tail artifact is a narrow pie-like region, which appears behind the shaft (see Figure 4.12). It is mainly caused by the repetitive sound reflections inside the needle. Any sound wave that enters in the shaft, starts reflecting back and forth inside the needle while loosing some fraction of itself at every bounce. Since these consecutively escaped sound in time is interpreted as coming from different depths by the ultrasound hardware, the comet tail forms and it consists of repetitive bright and dark stripes. If the speed of sound were the same in steel and tissue, the bright stripe separation would be twice the needle thickness. However, because of the higher speed of sound in steel, their separation becomes less than that.


Figure 4.12: Comet tail and multiple reflections artifacts of the needle.
Another artifact on the comet tail arises from the high reflectivity of the shaft, which is caused by the relatively high acoustic impedance of the needle with respect to the tissue ( $z_{\text {tissue }} \approx$ $1.54 \times 10^{6}$ Rayles, $z_{\text {steel }} \approx 47 \times 10^{6}$ Rayles, where Rayle is $\mathrm{kg} / \mathrm{m}^{2} \mathrm{~s}$ ). That creates multiple reflections artifacts, where obscure copies of the shaft are seen at the integer multiples of the actual shaft depth. This is caused by the sound that reflects back and forth between the needle and the probe. When the initial needle reflection hits the probe surface, most of it reflects back to the tissue medium perpendicularly, and it is then reflected from the needle surface once more. This second reflection and possible further ones generate the multiple less-bright shaft looks behind the actual one (see C4.5 in Figure 4.12).

Considering the above phenomena, the factors affecting the look of the needle shaft on the image can be categorized in two. The first is the distance from the probe to the shaft. This mainly determines the brightness and the radius of the ellipsoid denoting the shaft, and the separation between multiple reflections. It also indirectly affects the brightness of the comet tail and multiple reflections. The second factor is the angle between the horizon and a virtual line that is perpendicular to the probe surface and passing through the needle shaft (for instance, any point on column $D$ has this angle of instance equal to $90^{\circ}$ ). This angle is defined on the US image plane and it determines the orientation of the comet tail and the ellipsoid axis. Thus, the comet tail extends on this virtual line and the longer axis of the ellipsoid lies perpendicular to
this line.
For needle shaft synthesis, real needle ultrasound images were used. They were acquired at the hospital in a phantom using the same ultrasound system and probe settings that are used during the procedure. An acquired image can be seen in Figure 4.13(a). First its background was removed by subtracting the image of the region before needle insertion, then the remaining noise was eliminated using an adaptive thresholding filter. The processed "clean" version is shown in Figure 4.13(b).


Figure 4.13: Needle shaft.
In the code, we are only simulating the ellipsoid overlay at the proper position and orientation. Although multiple reflections are relatively easy to implement, comet tail synthesis needed further in-depth study. Since this thesis is not focused on US artifact modeling, better artifact synthesis schemes are yet to be researched in future work. For the shaft image, the position where the needle intersects the imaging plane is easily found using the needle knots just before and just after the imaging plane. This intersection gives where the needle should be positioned on the TRUS. With this information and using the priori-known coordinates of the probe, the angle of incidence can be calculated. The cleaned shaft image is then rotated accordingly and its brightness is adjusted to match with the TRUS image. Since adaptive thresholding filter leaves sharp edges, it is also smoothed with a $7 \times 7$ Gaussian filter. Finally, it is added on to the synthesized ultrasound image where its brightest pixel would correspond to the desired shaft center coordinate. The final generated image can be seen in Figure 4.11(c) with the overlayed needle between $2 B$ and 2.5 .

### 4.8 Discussion

In this chapter, two methods for generating TRUS images of soft tissue, the deformation of which was given by FEM, have been presented. The first one (in Section 4.4) has an advantage of precomputing basis functions and tagging the initially known voxels with the FEM elements they reside in. However, its interpolation requires many geometrical computations which would make its implementation slow. Therefore, the second one (in Section 4.4) was opted for. It makes use of the well-known regular-grid interpolations by transforming the plane of interest into the initial undeformed mesh.

Spatial optimizations (e.g., the displacements of some are computed, and the rest is approximated) or temporal estimations (e.g., a voxel stays mostly in the same element in-between iterations) can be applied. For the first presented method (namely direct method), there exists approximations for the interpolation. These should be addressed in the future work to decrease the running time of this synthesis process down to $\approx 0.1 \mathrm{~s}$ to be able to generate real-time TRUS images (original rate is 12 Hz ). Using 3D ultrasound would also give better resolution between slices.

The computational times in the end showed that the FEM-based transformations take in the order of minutes whereas interp3 function could interpolate in seconds (for linear $\approx 0.6 \mathrm{~s}$, for cubic $\approx 3 \mathrm{~s}$ ). The cause of the slow FEM-based transformation was its running on all the points $(\approx 1,400$ pixels). Approximations can be used for increasing this performance.

The negative points on the speed side were compensated on the quality side. The main problem could be the discontinuities in the image. The qualitatively comparisons assured the smoothness and realism of these generated images. However, a user study with sonographers would give a better idea.

The needle shaft appearance has also been simulated on a generated TRUS image. This was a preliminary attempt to show the possibility of synthesizing the needle tip, brachytherapy seeds, and their artifacts. It was seen that there were no previous studies on this. In this work, the feasibility of overlaying artificial images, such as the shaft reflection, without disturbing the realism of the underlying image has been studied. It has given promising results, however, once more modalities (e.g., artifacts) are added, the issue of validation has to be addressed. The current needle overlay can be validated through the following scheme: in the same phantom which the initial needle image was taken, other images at different angles can be taken and these can be compared with the needle overlayed ones generated by the algorithm above.

## Chapter 5

## Implementation

In this chapter, the implementation of the concepts given throughout the thesis is summarized. Although the FEA and ultrasound synthesis implementations below are currently not working in real-time, during their design and programming, their portability to a real-time system was given the utmost priority.

### 5.1 Program Overview

The tissue interaction is handled by a C code. This C-MEX structured file can be compiled in Matlab so that when it is called as a function with the state variables such as the needle base position, it will return the feedback values such as the needle base forces. Thus, a simulation of the system is possible such that the above function can be called repetitively while modifying the variables representing the scene change or disturbance. Each call represents an iteration cycle of a real-time haptic control loop.

At each iteration of this code, first the passed values and pointers to arrays and matrices are checked for any discrepancy among them. For an average iteration where the needle is in the tissue, the following operations are performed in the given order (they are detailed later in the chapter):

1. Transform the needle knot positions from the local needle frame into the world frame; find the orientations of the needle segments; and calculate the rotations from the previous iteration.
2. If a new tissue element is intercepted, modify the nominal mesh so that the closest node would be on the needle and add this node to the working system matrix.
3. Considering the needle direction, insertion/retraction, and the forces on the working nodes, decide the finite state on each working node and if the node slips off the tip, remove it from the working system matrix.
4. Set the constraints of each working node depending on its stick-slip state.
5. Solve the working subsystem to find the displacements/forces on the working nodes.
6. Calculate the relative position of each working node on the needle and determine the tissue contact forces (shaft friction and tip puncture) on them to be used in the next iteration.
7. Project the tissue forces of the working nodes onto the knots of the needle model which is in a separate frame by itself.
8. Sum up the working node forces to find out the force feedback to be applied at the needle base on the user's hand.
9. Solve the needle model for its knot displacements in its own local frame.
10. Multiply the working node forces (since forces at other nodes are zero) by the nominal inverse stiffness matrix to find the displacements of all the nodes.

### 5.2 Simulation Structure

For simulation of the insertion the C-MEX file is compiled into a DLL executable. Through an interface designed in Matlab, the user can input the next destination of the needle base. Then the Matlab program calculates a linear trajectory and runs a loop, in which first the needle base position is updated and then the executable is called. The tissue and the needle vertices are used by the interface to draw them on screen. Since Matlab drawing functions are relatively slow, the drawing is done at every $N^{t h}$ iteration, where $N$ can be set through the interface. Figure 5.1 shows screenshots of the interface.

At the beginning of the simulation, the matrices modeling the tissue and the needle are initialized. Since the tissue model takes up to 30 sec to compute, when tissue matrices are generated once, they are saved in a file to be loaded in further runs. Then the initial values of the variables are defined according to the initial setup of the volume. Finally the iterative scheme in Section 5.5 is started.

Currently the TRUS image synthesis is implemented separately and it is initiated by a button in the interface. This generates a slice at a constant depth in order of minutes. That is why the current implementation is not suitable for a visual TRUS rate of 12 Hz . Further approximations and optimizations have been suggested in the related chapter (Chapter 4).

### 5.3 Tissue Model Initialization

The variables are named here to make them easier to follow on Table 5.1 where $n$ is the number of nodes, $e$ is the number of tetrahedra, $t$ is the number of materials, $b$ is the number of constrained nodes, and $v$ is the maximum node valence.

First, node numbers (NodeList) and material types (ElemType) of tetrahedra elements, and vertex positions (Xn) are read from the mesh file generated by GiD. Using the element types, the nodes inside the regions and on the interfaces are also set a type (NodeType). Then each node


Figure 5.1: Matlab interface running the simulation.
is assigned a friction constant and a puncture threshold (Fd). Tissue parameters (TissuePar) define Young's modulus $E$ and Poisson's ratio $\nu$ of prostate and surrounding soft tissue.

During the simulation, at many points the elements connected to a node are needed. Therefore, a node-element connectivity list (NEconn) is generated offline to avoid traversing the element list each time.

Before starting the stiffness matrix creation, the nodes (Nfixed) which are constrained in space are defined. In the current version of the simulation, the entire volume is fixed in space only from a couple of nodes on the surface of the prostate above its apex (denoted by the colored patch on the prostate surface in Figure 5.1). This is to represent the stiff tissue connecting it to the pubic arch and to display its effect on the torquing of the prostate as the needle is inserted. If the pubic arch were segmented, all of its nodes would have been used as fixed displacement boundary conditions since a major skeletal bone can be assumed a mechanical ground. Segmenting that other anatomy would make the simulation more realistic.

| Name | Dimension | Content |
| :--- | :---: | :--- |
| Xn | $n \times 3$ | Nominal positions of tissue nodes. (its order sets the global num- <br> bering) |
| NodeList | $e \times 4$ | Indices of each element's corner nodes. (sets element numbering) |
| ElemType | $e$ | The tissue content of each element. |
| Nodetype | $n$ | The tissue type at each node. |
| Fd | $n \times 2$ | Shaft friction and tip puncture forces for each node. |
| TissuePar | $t \times 2$ | Parameters $(E, \nu)$ of each tissue type. |
| NEconn | $n \times v$ | The indices of elements surrounding each node. |
| Nfixed | $b$ | The indices of the nodes with displacement boundary condition. |
| Kinv | $3 n \times 3 n$ | The inverse global stiffness matrix for tissue FEM. |

Table 5.1: Tissue FEA variables.

One of the last offline initialization steps is the computation of the inverse global stiffness matrix $K^{-1}$ as in Algorithm 5.1. On Line 11, this huge matrix is inverted using the Matlab's inv function.

```
Algorithm 5.1 Creating the inverse stiffness matrix offline.
    function CreateInvStiffMat(TissuePar, ElemType, Xn, NodeList, Nfixed)
        TissuePar \((E, \nu) \rightarrow\) Lamé constants \((\lambda, \mu) \rightarrow \mathrm{C} \quad \triangleright\) (A.12) and (A.11)
        Kglobal \(\leftarrow 0_{3 n \times 3 n}\)
        for \(i \leftarrow 1\) to \(e\) do
            Find element volume \(v \quad \triangleright(3.1)\)
            Find \(B^{e}\)
                                \(\triangleright\) (A.18) and (A.24)
            \(K_{12 \times 12}^{e} \leftarrow v B^{e T} C B^{e} \quad \triangleright(\mathrm{~A} .30)\)
            Add \(K_{12 \times 12}^{e}\) onto the corresponding elements of Kglobal
            \(\triangleright\) (A.32)
        for all \(i \in\) Nfixed do
            the rows/columns \(\{3 i-2,3 i-1,3 i\}\) of Kglobal \(\leftarrow I_{3 \times 3} \quad \triangleright(\) A.33 \()\)
        \(K^{-1} \leftarrow(\text { Kglobal })^{\prime}\)
```


### 5.4 Needle Model Initialization

The needle model is also created before the simulation starts. For this, the variables in Table 5.2 are set. First the mesh is generated by setting node positions (NXn) and elements (NNodeList) in a loop, according to the mesh structure desired. A simple or a symmetric mesh can be chosen as presented in Section 3.5. For the final simulation, the symmetric structure is used. Then its stiffness matrix is calculated using the same function in Algorithm 5.1, the needle's base end is positionally constrained, and the inverse stiffness matrix (NKinv) is calculated.

| Name | Dimension | Content |
| :--- | :---: | :--- |
| NXn | $\hat{n} \times 3$ | Nominal positions of needle nodes. |
| NNodeList | $\hat{e} \times 4$ | Indices of each needle element's corner nodes. |
| NKinv | $3 \hat{n} \times 3 \hat{n}$ | The inverse global stiffness matrix for needle FEM. |

Table 5.2: Needle FEA variables.

### 5.5 An Iteration

The flowchart of a code iteration is given in Figure 5.2. Its functions are detailed as follows:


Figure 5.2: A main iteration flowchart of needle interacting with the tissue.

- Locateneedle: The needle base position and orientation are taken from the graphical interface (for haptics they will be read from the haptic simulator device). Adding $U_{k}$, the needle knot world-frame positions are computed. Comparing the tip coordinate with the previous iteration's, the needle motion direction (advance/retract) is set as well. (This direction is used for the stick-slip automatons.)
- SetConstraintAngles: As also given in Algorithm 5.2, first using the needle knot positions, the needle segment orientations are calculated ${ }^{1}$ geometrically (atan2 function was employed). These are necessary for local frame updates in Section 2.4. Then, for each

[^7]contact node $j$ the cumulative rotation matrix $R_{T N, i}^{j}$ from the needle frame to the tissue frame is found, where $i$ denotes the current iteration. Finally, an incremental rotation matrix $R_{T_{i} T_{i-1}}^{j}$ is set by calculating $R_{T N, i}^{j}\left(R_{T N, i-1}^{j}\right)^{-1}$. The incremental rotation matrices are used for $\boldsymbol{K}_{W}$ update at each iteration and the cumulative one is used when a new node is added. To add a node, $\boldsymbol{K}_{H}$ matrix is reformed from $K^{-1}$ which resets the orientations to the initial geometry. Therefore, the initial rotated working matrix $\boldsymbol{K}_{W}$ is regenerated by updating $\boldsymbol{K}_{H}$ using cumulative rotations.

```
Algorithm 5.2 Constraint angle calculation for a contact node.
    Find \((\phi, \theta, \psi)_{\text {segment }} \quad \triangleright\) found geometrically
    for all \(j \in\) contact nodes do Find \(R_{T N, i}^{j}\)
    \(R_{T_{i} T_{i-1}}^{j} \leftarrow R_{T N, i}^{j}\left(R_{T N, i-1}^{j}\right)^{-1}\)
```

- CheckNewElementIntercept: If the element that the tip is in has no (working) nodes on the needle, then its closest corner is needed to be added on the needle after a remeshing stage. Otherwise the flow jumps to contact node servicing.
- ModifyMesh: This implements the steps 1 to 3 in Section 2.5. First, the (approximate) nominal coordinate for the tip is found, then the boundary conditions are switched on $\boldsymbol{K}^{-1}$ for the surrounding nodes. After that, the current surrounding elements are removed by subtracting their $K^{e}$ and the new elements are added similarly. Finally, the boundary conditions are switched back. In the end, Xn is updated with $P^{\prime}$ to reflect the coordinate change.
- AddNode: The new node $j$ is added to the list of contact nodes and its variables (e.g., the rotation matrices $j$ and $R_{T_{i} T_{i-1}}^{j}$ ) are initialized. Rows/columns corresponding to the contact nodes are copied from $\boldsymbol{K}^{-1}$ to the condensed system matrix $\boldsymbol{K}^{H}$. Finally, adjusting the LCFs using cumulative rotations $R_{T N, i}$ and switching the boundary conditions ( $x, y, z$ for stuck; only $x$ and $z$ for slip), the working matrix $\boldsymbol{K}_{W}$ is obtained.
- ServiceFiniteStateMachine: If a working node slips off the tip, its corresponding rows/columns are removed from $\boldsymbol{K}_{W}$. The stick-slip states are serviced as given in Algorithm 5.3.
- SetConstraints: Each contact node's world-frame position is found using the orientations and the knots of the segment that it resides on. For stuck nodes, corresponding displacements applied on three $\underline{\boldsymbol{y}}$ rows/columns; for slipping nodes, two displacements and one force (friction threshold) are applied. Then, the working subsystem is solved and the forces/displacements are arranged into separate vectors.

For each contact node, an axial friction force is defined according to the tissue material and the length of needle that this contact node is representing. In other words, the friction on

```
Algorithm 5.3 Finite state machine servicing for a contact node.
    if stuck then
        Update LCF for stuck using \(R_{T_{i} T_{i-1}}^{j}\)
        if "force along y" > "friction threshold" then
            Boundary condition change along y
            Set its state to "slip"
    if slip then
        if TipMotion \(_{i} \neq\) TipMotion \(_{i-1}\) then
            Boundary condition change along y
            Update LCF for stuck using \(R_{T_{i} T_{i-1}}^{j}\)
            Set its state to "stuck"
        else
            Update LCF for slip using \(R_{T_{i} T_{i-1}}^{j}\)
the needle surface between this node and the next contact node (or the tip) is accumulated and applied as a friction force on this contact node. It is used both as a force boundary condition of a slipping node and a static friction threshold for a stuck node (such that the node will start slipping past that limit).
- ProjectForces: This first transforms the contact node forces into needle frame (using \(\left.R_{T N, i}\right)\) and then distributes them onto the needle knots linearly as in Section 2.8.
- IntegrateForces: This sums up the forces and the torques of contact nodes on the base (see Section 2.9). This is to be applied by the haptic device on the user hand.

If a valid model for tip bevel deflection is found, it will probably be in the form of a pure tip force. If this is the case, this force can be added onto the needle model at this point of the iteration cycle.
- SolveNeedle: After distributing the knot forces (dividing them by 3), the needle model is iteratively solved (FEA) for the needle displacements using the applied forces as shown in Section 2.1. At each iteration a velocity gain is applied on the displacements such that \(\hat{u}_{i}=\hat{u}_{i-1}+k \Delta \hat{u}\). For stability, this gain needs to be adjusted according to the Young's modulus and Poisson's ratio of the needle. This solution converges within \(1 \%\) margin in less than 10 iterations for proper values and acceptable forces/deflections.

Finally, the displacements and positions of all the nodes are calculated as mentioned in Section 2.7. Although it is not crucial to compute them at haptic refresh rate, the current implementation is also using some of the deformed node positions in various operations (e.g., penetration check in CheckNewElementIntercept). Moving this step to a second (visualization) computer and its drawbacks are discussed in Section 5.7.

\subsection*{5.6 Notes}

An advantage of the C-MEX file is that it can easily be compiled and loaded onto a target machine in the future without the need for many changes. Here target machine refers to a computer that can handle fast haptic loops while communicating with optical encoders and motor driving amplifiers at a fast and reliable rate by usually running a real-time operating system (e.g., VxWorks \({ }^{\circledR}\) ). That structure allows the use of WindRiver's Tornado \({ }^{\circledR}\) development environment and the control of runtime variables in Matlab Simulink \({ }^{\circledR}\) through the Real-Time Workshop \({ }^{\circledR}\) toolbox, in which the author has experience with.

For matrix operations, the functions such as matrix multiplication and rank 3-4 inversion are included in the code. However, FEA matrix operations were not optimized, i.e., although there are many zero elements in the matrices \(B^{e}\) and \(C\), they are actually multiplied element by element in the code. Skipping zero terms would could speed up the algorithms, but the current bottleneck is the \(\mathcal{O}\left(n^{2}\right)\) operations (mesh adaptation), therefore all the other optimizations are given a lower priority. The presented \(\mathcal{O}\left(m^{2} n\right)\) approach decreased the \(\mathcal{O}\left(m n^{2}\right)\) mesh adaptation time of two sample meshes with 836 nodes and 240 nodes from \(\approx 13 \mathrm{~s}\) and \(\approx 1 \mathrm{~s}\) down to \(\approx 1.25 \mathrm{~s}\) and \(\approx 0.02 \mathrm{~s}\), respectively. The memory requirements for \(\boldsymbol{K}^{-1}\) storage are 50 MB and 4 MB , respectively.

Figure 5.3 shows the needle base reaction force and the time requirement of each iteration during the insertion shown in Figure 5.1. The needle is modeled using a symmetric mesh (shown in Section 3.5) having 32 elements and it is pushed at a constant velocity of \(0.1 \mathrm{~mm} /\) iteration. This would correspond to a \(5 \mathrm{~cm} / \mathrm{s}\) speed in a haptic implementation running at 500 Hz . In Figure \(5.3(\mathrm{~b})\), note that the computations take \(2-3 \mathrm{~s}\) during mesh modification using \(\mathcal{O}\left(m n^{2}\right)\). The effect of the different stiffness and the friction of prostate tissue can be observed in Figure 5.3(a) around the \(380^{\text {th }}\) iteration.


Figure 5.3: The simulation forces and timings.

\subsection*{5.7 Discussion}

In this chapter, the implementation of extensions to DiMaio's work [40] have been described. One major change was extending the algorithms to 3D, which generated challenges in local frame handling and meshing (needle asymmetry discussed in Chapter 3). The other main improvement was the custom meshing. The data structures and algorithms were set to accommodate any number of nodes and elements having arbitrary connectivity (without dependency on maximum node valence) so that high-level meshing can be used in order to have a more accurate model with high-quality elements and a small number of nodes.

Although the current code is not fast enough, employing a node snap (where the closest node is constrained on the needle without remeshing, which is also implemented in DiMaio's needle insertion simulator) instead of remeshing would let it run at a haptic rate ( \(>200 \mathrm{~Hz}\) ). Although the needle flexibility causes problems due to force spikes, these issues can be attempted using force smoothing techniques.

In order to make this implementation faster, two burdens, namely the mesh adaptation and visualization calculations, can be removed from the haptic loop together. They can be both moved to a separate computer (let us call it TRUS computer), which holds the \(\boldsymbol{K}^{-1}\) matrix. When a node addition is required, the TRUS computer can send the condensed matrix \(\left(\boldsymbol{K}_{H}\right)\) to the haptic controller which processes it to obtain \(\boldsymbol{K}_{W}\). The remeshing can also be implemented such that the new node \(P^{\prime}\) found by the haptic controller can be sent to the TRUS computer, which can then update the \(\boldsymbol{K}^{-1}\) matrix (and sends back the new condensed matrix). This structure has two flaws: the time lag of the remeshing and the node positions for new element interception check. The former one refers to the time passes on the haptic side between the element interception and the return of \(\boldsymbol{K}_{H}\). Nonetheless, the haptic loop can continue handling the interaction and apply the changes when the matrix is available (maybe using some additional force smoothing). The latter refers to the need of the penetration check algorithm for the deformed positions of the surrounding nodes. Since there only a few of those nodes, their positions can be sent back by the TRUS computer.

\section*{Chapter 6}

\section*{Conclusion}

\subsection*{6.1 Summary and Contributions}

A prostate brachytherapy simulator design has been presented. The design consists of two major subsystems, a needle-tissue interaction model and a TRUS image synthesizer. The specific contributions of this thesis can be summarized as follows:

3D Extension: In this thesis, the concepts which were proposed in prior work [40] are extended to 3D. The presented system is the first physically-based 3D interaction model of a flexible needle insertion into a soft deformable body. Existing literature has presented heuristic approaches [20, 21], 2D FEM implementations [9, 40, 36], and a sample insertion of a rigid needle into nonlinear 3D FEM [97]. Nevertheless, the only physically-based 3D work by [97] utilized a non-conforming FEM, whereas the proposed method successfully couples a conforming quasi-static linear FEM of the tissue with the flexible needle.
"Effective" Meshes: Using the presented scheme, a 3D volume can be meshed with conforming region boundaries on surfaces generated from contours on parallel slices. This method achieves relatively high-quality meshes which are also suitable for fast haptic implementations of the FEM.

Brachytherapy Simulation: A prostate mesh is generated using the above mentioned method. Incorporation of this mesh into the proposed interaction model has resulted in the first physically-based 3D needle insertion simulation of a brachytherapy procedure. The issues related to the haptic implementation have been addressed through this simulation which is currently running in a Matlab interface.

TRUS Synthesis: The feasibility of TRUS generation as a visual feedback modality in the above mentioned simulation has been studied. To the best of our knowledge, the presented technique for ultrasound synthesis from a deformed volume with deformations defined by a finite element model is a novel contribution to this field. The speed considerations have also been addressed and a sample case of needle overlay has been presented.

\subsection*{6.2 Future Work}

Segmenting more anatomical structures of the region and incorporating them into the simulator will increase the realism. Most importantly the pubic arch is needed as a fundamental boundary condition. The bladder is also important as a fluid pocket. The fatty and muscle tissue and the urethra can also be added to increase the realism. Moreover, a finer meshing of the tissue regions, where the needle passes frequently or seeds are implemented (namely PTV), might increase the simulation accuracy. On the other hand, the values which were set empirically for the presented mesh (e.g., preferred element sizes) can be studied for their effects on the final performance so that these parameters can be theoretically bounded by means of expected accuracy.

The linear quasi-static model should be verified to see how well it represents the reaction forces and deformations of the prostate region. Before an in vivo validation, 3D deformations of tissue-like phantoms and in vitro tissue samples can be used. Depending on the performance of these deformation models, nonlinear and dynamic models may be attempted next. In particular dynamic characteristics may bring more realism to the simulator (e.g., the inertia of the tissue can then be felt). Other elasticity models may be attempted as well.

The demonstrated iterative solution of the quadratic-strain needle model does not converge in some ranges of its elasticity parameters. Other solution methods which were shown to converge require large matrix inversions \(\left(\hat{\mathbf{K}}_{q}^{-1}\right.\) which is rank \(\left.\approx 50-100\right)\) at each iteration. Thus, before a haptic implementation, this may need to be addressed in order to prevent instability. The feasibility of other approaches (such as mass-spring models) for needle bending can also be investigated. The needle tip bevel effect should also be studied and incorporated into the simulator. Finding the bevel deflection through model fitting and parameter optimization using experimental insertions may be a promising approach.

The TRUS synthesis rate has to be brought up to 12 Hz for real-time implementation. Although the current computation of TRUS generation is rather slow, there are numerous approximation schemes that can be applied to improve performance. For instance, calculating deformation for one every 10 pixels in each direction and then interpolating the deformations of the rest would increase speed by 2 orders of magnitude and drop the computation time down to the order of seconds. A combination of different approaches can be utilized for interpolation. For example, different regions of the images can be interpolated using different methods. However, these patches might be hard to fuse realistically.

The proposed simulator or a simplified version of it can be implemented on a haptic device in order to address possible implementation problems. Two computer or dual processor implementations can be considered. Priority scheduling of VxWorks can also be utilized for background processing of low-priority tasks. The literature addresses similar problems and implementations extensively and a suitable solution can be extracted from these related studies.

\section*{Bibliography}
[1] National Cancer Institute of Canada, "Canadian cancer statistics 2004," 2004. http://www.cancer.ca.
[2] W. M. Butler, "Review of modern prostate brachytherapy," in Proceedings of the 222nd Annual EMBS International, (Chicago, IL), pp. 748-752, 2000.
[3] R. Taschereau, J. Pouliot, J. Roy, and D. Tremblay, "Seed misplacement and stabilizing needles in transperineal permanent prostate implants," Radiotherapy and Oncology, vol. 55, no. 1, pp. 59-63, 2000.
[4] B. J. Davis, R. R. Kinnick, M. Fatemi, E. P. Lief, R. A. Robb, and J. F. Greenleaf, "Measurement of the ultrasound backscatter signal from three seed types as a function of incidence angle: Application to permanent prostate brachytherapy," International Journal of Radiation Oncology*Biology*Physics, vol. 57, no. 4, pp. 1174-1182, 2003.
[5] P. J. Gorman, T. M. Krummel, R. Webster, M. Smith, and D. Hutchens, "A prototype haptic lumbar puncture simulator," in Proceedings of \(8^{\text {th }}\) Medicine Meets Virtual Reality (MMVR), pp. 263-274, 2000.
[6] F. S. Azar, D. N. Metaxas, and M. D. Schnall, "Methods for modeling and predicting mechanical deformations of the breast under external perturbations," in Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp. 1267-1270, 2001.
[7] S. Miller, C. Jeffrey, J. Bews, and W. Kinsner, "Advances in the virtual reality interstitial brachytherapy system," in IEEE Canadian Conference on Electrical and Computer Engineering, (Edmonton, Alberta), 1999.
[8] J. K. Hahn, M. J. Manyak, G. Jin, D. Kim, J. Rewcastle, S. Kim, and R. J. Walsh, "Cryotherapy simulator for localized prostate cancer," in Proceedings of \(10^{\text {th }}\) Medicine Meets Virtual Reality (MMVR), 2002.
[9] R. Alterovitz, K. Goldberg, J. Pouliot, R. Taschereau, and I.-C. Hsu, "Sensorless planning for medical needle insertion procedures," in IEEE/RSJ International Conference on Intelligent Robots and Systems, (Las Vegas, NV), 2003.
[10] G. Picinbono, H. Delingette, and N. Ayache, "Non-linear and anisotropic elastic soft tissue models for medical simulation," in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1370-1375, 2001.
[11] Immersion, "Endoscopy, laparoscopy, intra-vascular simulators," 2004. http://www.immersion.com.
[12] Simbionix, "Endoscopy, laparoscopy, intra-vascular simulators," 2004. http://www.simbionix.com.
[13] Mentice, "Laparoscopy simulator," 2004. http://www.mentice.com.
[14] D. Terzopoulos and K. Waters, "Physically-based facial modeling, analysis and animation," Visualization and Computer Animation, vol. 1, no. 2, pp. 73-80, 1990.
[15] M. Downes, M. C. Cavusoglu, W. Gantert, L. W. Way, and F. Tendick, "Virtual environments for training critical skills in laparoscopic surgery," in Proceedings of \(6^{\text {th }}\) Medicine Meets Virtual Reality (MMVR), (San Diego, CA), pp. 316-322, 1998.
[16] S. P. DiMaio and S. E. Salcudean, "Needle insertion modeling and simulation," IEEE Transactions on Robotics and Automation, vol. 19, no. 5, pp. 864-875, 2003.
[17] L. Hiemenz, J. McDonald, D. Stredney, and D. Sessana, "A physiologically valid simulator for training residents to perform an epidural block," in Proceedings of the \(15^{\text {th }}\) Southern Biomedical Engineering Conference, pp. 170-173, 1996.
[18] G. Hirota, S. Fisher, and A. State, "An improved finite element contact model for anatomical simulations," The Visual Computer, vol. 19, no. 5, pp. 291-309, 2003.
[19] H. Fuchs, A. State, E. D. Pisano, W. F. Garrett, G. Hirota, M. Livingston, M. C. Whitton, and S. M. Pizer, "Towards performing ultrasound-guided needle biopsies from within a head-mounted display," in Proceedings of Visualization in Biomedical Computing, (Hamburg, Germany), pp. 591-600, 1996.
[20] A. N. Kimura, J. Camp, R. Robb, and B. Davis, "A prostate brachytherapy training rehearsal system - simulation of deformable needle insertion," in Proceedings of Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp. 264-271, 2002.
[21] X. Wang and A. Fenster, "A haptic-enhanced 3D real-time interactive needle insertion simulation for prostate brachytherapy," in Prooceedings of SPIE Medical Imaging 2004: Visualization, ImageGuided Procedures, and Display, vol. 5367, pp. 781-789, 2004.
[22] C. M. Schneider, A. M. Okamura, and G. Fichtinger, "A robotic system for transrectal needle insertion into the prostate with integrated ultrasound," in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), (New Orleans, LA), pp. 365-370, 2004.
[23] Z. Wei, G. Wan, L. Gardi, D. B. Downey, and A. Fenster, "Robotic aided 3D TRUS guided intraoperative prostate brachytherapy," in the Proceedings of SPIE Medical Imaging 2004: Visualization, Image-Guided Procedures, and Display, vol. 5367, pp. 361-370, 2004.
[24] S. F. F. Gibson and B. Mirtich, "A survey of deformable modeling in computer graphics," Tech. Rep. TR-97-19, MERL - Mitsubishi Electric Research Laboratory, 1997.
[25] S. F. F. Gibson, "3D ChainMail: A fast algorithm for deforming volumetric objects," in Proceedings of 1997 Symposium on Interactive 3D Graphics, pp. 149-154, 1997.
[26] K. Waters, "A muscle model for animating three-dimensional facial expression," Computer Graphics, vol. 21, no. 4, pp. 17-24, 1987.
[27] E. Promayon, P. Baconnier, and C. Puech, "Physically-based deformations constrained in displacements and volume," Computer Graphics Forum (EUROGRAPHICS'96), vol. 15, no. 3, pp. 155-164, 1996.
[28] D. L. James and D. K. Pai, "Artdefo: Accurate real time deformable objects," in Proceedings of the \(26^{t h}\) Annual Conference on Computer Graphics and Interactive Techniques, pp. 65-72, 1999.
[29] D. L. James and D. K. Pai, "Dyrt: Dynamic response textures for real time deformation simulation with graphics hardware," ACM Transactions on Graphics (SIGGRAPH Proceedings), vol. 21, no. 3, pp. 582-585, 2002.
[30] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformable models," Computer Graphics, vol. 21, no. 4, pp. 205-214, 1987.
[31] S. Cotin, H. Delingette, and N. Ayache, "Real-time elastic deformations of soft tissues for surgery simulation," IEEE Transactions On Visualization and Computer Graphics, vol. 5, no. 1, pp. 62-73, 1999.
[32] Y. Zhuang and J. Canny, "Real-time simulation of physically realistic global deformation," in SIGGRAPH99 Sketches and Applications, (Los Angeles, CA), 1999.
[33] Y. Zhuang, Real-time simulation of physically realistic global deformations. PhD thesis, CS, University of California Berkeley, 2000.
[34] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr, "Dynamical real-time deformations using space \& time adaptive sampling," ACM Transactions on Graphics (SIGGRAPH Proceedings), pp. 31-36, 2001.
[35] X. Wu, M. S. Downes, T. Goktekin, and F. Tendick, "Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes," Computer Graphics Forum (EUROGRAPHICS'01), vol. 20, no. 3, 2001.
[36] H.-W. Nienhuys, Cutting in deformable objects. PhD thesis, Utrecht University, 2003.
[37] A. E. Kerdok, S. M. Cotin, M. P. Ottensmeyer, A. M. Galea, R. D. Howe, and S. L. Dawson, "Truth cube: Establishing physical standards for soft tissue simulation," Medical Image Analysis, vol. 7, pp. 283-291, 2003.
[38] S. P. DiMaio and S. E. Salcudean, "Simulated interactive needle insertion," in Proceedings of the \(10^{\text {th }}\) Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS), pp. 344-351, 2002.
[39] J. Berkley, G. Turkiyyah, D. Berg, M. A. 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, no. 3, pp. 314-325, 2004.
[40] S. DiMaio, Modelling, Simulation and Planning of Needle Motion in Soft Tissues. PhD thesis, ECE, University of British Columbia, 2003.
[41] M. Bro-Nielsen and S. Cotin, "Real-time volumetric deformable models for surgery simulation using finite elements and condensation," Computer Graphics Forum (EUROGRAPHICS'96), vol. 15, no. 3, pp. 57-66, 1996.
[42] A. H. Gosline, S. E. Salcudean, and J. Yan, "Haptic simulation of linear elastic media with fluid pockets," in Proceedings of the \(12^{\text {th }}\) Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS), pp. 266-271, 2004.
[43] 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.
[44] C. Simone and A. M. Okamura, "Modeling of needle insertion forces for robot-assisted percutaneous therapy," in Proceedings of IEEE International Conference on Robotics and Automation, (Washington, DC), 2002.
[45] H. Kataoka, T. Washio, K. Chinzei, K. Mizuhara, C. Simone, and A. M. Okamura, "Measurement of the tip and friction force acting on a needle during penetration," in Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp. 216-223, 2000.
[46] M. D. O'Leary, C. Simone, T. Washio, K. Yoshinaka, and A. M. Okamura, "Robotic needle insertion: Effects of friction and needle geometry," in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), (Taipei, Taiwan), pp. 1774-1780, 2003.
[47] A. M. Okamura, C. Simone, and M. D. OLeary, "Force modeling for needle insertion into soft tissue," IEEE Transactions on Biomedical Engineering, vol. 51, no. 10, pp. 1707-1716, 2004.
[48] S. P. DiMaio and S. E. Salcudean, "Needle steering and model-based trajectory planning," in Proceedings of Medical Image Computing and Computer-Assisted Intervention (MICCAI), (Montreal, QC), 2003.
[49] R. J. WebsterIII, N. J. Cowan, G. Chirikjian, and A. M. Okamura, "Nonholonomic modeling of needle steering." to appear at \(9^{t h}\) International Symposium on Experimental Robotics, June 2004.
[50] R. Alterovitz, K. Goldberg, and A. Okamura, "Planning for steerable bevel-tip needle insertion through 2D soft tissue with obstacles." to appear at "IEEE International Conference on Robotics and Automation (ICRA) 2005", April 2005.
[51] B. Cutler, J. Dorsey, L. McMillan, M. Müller, and R. Jagnow, "A procedural approach to authoring solid models," ACM Transactions on Graphics (SIGGRAPH Proceedings), vol. 21, no. 3, pp. 302-311, 2002.
[52] J. F. O'Brien, A. W. Bargteil, and J. K. Hodgins, "Graphical modeling and animation of ductile fracture," ACM Transactions on Graphics (SIGGRAPH Proceedings), vol. 21, no. 3, pp. 291-294, 2002.
[53] K. Dhinsa, C. Bailey, and K. Pericleous, "Investigation into the performance of turbulence models for fluid flow and heat transfer phenomena in electronic applications," in \(20^{\text {th }}\) Annual IEEE Semiconductor Thermal Measurement and Management (SEMI-THERM), pp. 278-285, 2004.
[54] C. Singh Verma and V. C. V. Rao, "Parallelization of finite volume computations for heat transfer application using unstructured mesh partitioning algorithms," in \(4^{\text {th }}\) International Conference on High Performance Computing, 1997.
[55] F. Ganovelli, P. Cignoni, C. Montani, and R. Scopigno, "Enabling cuts on multiresolution representation," The Visual Computer, vol. 17, no. 5, pp. 274-286, 2001.
[56] E. Grinspun, P. Krysl, and P. Schröder, "Charms: a simple framework for adaptive simulation," ACM Transactions on Graphics (SIGGRAPH Proceedings), vol. 21, no. 3, pp. 281-290, 2002.
[57] M. Müller and M. Teschner, "Volumetric meshes for real-time medical simulations," in Proceedings of \(7^{\text {th }}\) Workshop on Bildverarbeitung für die Medizin, (Erlangen, Germany), pp. 279-283, SpringerVerlag, 2003.
[58] P. Fleischmann, R. Kosik, and S. Selberherr, "Simple mesh examples to illustrate specific finite element mesh requirements," in \(8^{\text {th }}\) International Meshing Roundtable, pp. 241-246, 1999.
[59] R. V. Garimella and M. S. Shephard, "Boundary layer meshing for viscous flows in complex domains," in \(7^{\text {th }}\) International Meshing Roundtable, pp. 107-118, 1998.
[60] M. A. Yerry and M. S. Shephard, "Automatic three-dimensional mesh generation by the modified octree technique," International Journal For Numerical Methods in Engineering, no. 20, pp. 19651990, 1984.
[61] R. Löhner and P. Parikh, "Three-dimensional grid generation by the advancing front method," International Journal for Numerical Methods in Fluids, vol. 8, pp. 1135-1149, 1988.
[62] S. H. Lo, "Volume discretization into tetrahedra - II: 3D triangulation by advancing front approach," Computers and Structures, vol. 39, no. 5, pp. 501-511, 1991.
[63] B. N. Delaunay, "Sur la Sphere - vide. izvestia akademia nauk sssr," VII Seria, Otdelenie Matematicheskii i Estestvennyka Nauk, vol. 7, pp. 793-800, 1934.
[64] D. F. Watson, "Computing the delaunay tesselation with application to voronoi polytopes," The Computer Journal, vol. 24, no. 2, pp. 167-172, 1981.
[65] J. R. Shewchuk, "Tetrahedral mesh generation by delaunay refinement," in Symposium on Computational Geometry (SCG), (Minneapolis, MN), pp. 86-95, 1998.
[66] J. R. Shewchuk, "Triangle: Engineering a 2D quality mesh generator and delaunay triangulator," 1996. http://www.cs.cmu.edu/~quake/triangle.html.
[67] H. Borouchaki, F. Hecht, E. Saltel, and P. L. George, "Reasonably efficient delaunay based mesh generator in three dimensions," in \(4^{\text {th }}\) International Meshing Roundtable, pp. 3-14, 1995.
[68] P. L. George, F. Hecht, and E. Saltel, "Automatic mesh generator with specified boundary," Computer Methods in Applied Mechanics and Engineering, vol. 92, pp. 269-288, 1991.
[69] M. Bern and P. Plassmann, "Mesh generation," in Handbook of Computational Geometry (J.-R. Sack and J. Urrutia, eds.), Elsevier Science, 2000.
[70] S. J. Owen, "A survey of unstructured mesh generation technology," in rth International Meshing Roundtable, (Dearborn, MI), pp. 239-267, 1998. also available at http://www.andrew.cmu.edu/user/sowen/survey/.
[71] N. Molino, R. Bridson, J. Teran, and R. Fedkiw, "A crystalline, red green strategy for meshing highly deformable objects with tetrahedra," in \(12^{\text {th }}\) International Meshing Roundtable, pp. 103-114, 2003.
[72] T. R. Nelson, D. B. Downey, D. H. Pretorious, and A. Fenster, Three Dimensional Ultrasound. Lippincott Williams \& Wilkins, 1999. US.
[73] R. N. Rohling, A. H. Gee, and L. Berman, "A comparison of freehand three-dimensional ultrasound reconstruction techniques," Medical Image Analysis, vol. 3, no. 4, pp. 339-359, 1999.
[74] P. R. Detmer, G. Bashein, and T. Hodges, "3-D ultrasonic image feature location based on magnetic scan head tracking: In vitro calibration and validation," Ultrasound in Medicine and Biology, vol. 20, pp. 923-936, 1994.
[75] R. N. Rohling, A. H. Gee, and L. Berman, "Three-dimensional spatial compounding of ultrasound images," Medical Image Analysis, vol. 1, no. 3, pp. 177-193, 1997.
[76] D. F. Leotta and R. W. Martin, "Three-dimensional ultrasound imaging of the rotator cuff: Spatial compounding and tendon thickness measurement," Ultrasound in Medicine and Biology, vol. 25, no. 4, pp. 509-525, 2000.
[77] R. Franke, "Scattered data interpolation: Tests of some methods," Math Computation, vol. 38, no. 157, pp. 181-200, 2000.
[78] J. M. Thijssen and B. J. Oosterveld, "Speckle and texture in echography: Artifact or information?," in Proceedings of Ultrasonics Symposium, pp. 803-809, 1986.
[79] C. D. Barry, C. P. Allott, N. W. John, P. M. Mellor, P. A. Arundel, D. S. Thomson, and J. C. Waterton, "Scattered data interpolation: Tests of some methods," Ultrasound in Medicine and Biology, vol. 23, no. 8, pp. 1209-1224, 1997.
[80] R. N. Rohling, A. H. Gee, L. Berman, and G. Treece, "Radial basis function interpolation for freehand 3D ultrasound," in Proceedings of the International Conference on Image Processing in Medical Imaging, (Visegrad, Hungary), 1999.
[81] T. Roxborough and G. Nielson, "Tetrahedron based, least square, progressive volume models with applications to freehand ultrasound data," in Proceedings of IEEE Visualization, pp. 93-100, 2000.
[82] S. Meairs, J. Beyer, and M. Hennerici, "Reconstruction and visualization of irregularly sampled threeand four-dimensional ultrasound data for cerebrovascular applications," Ultrasound in Medicine and Biology, vol. 26, no. 2, pp. 263-272, 2000.
[83] H. Watanabe, H. Kaiho, M. Tanaka, and Y. Tersawa, "Diagnostic application of ultrasonography to the prostate," Invest. Ural., vol. 8, pp. 548-549, 1971.
[84] S. Tong, D. B. Downey, H. N. Cardinal, and A. Fenster, "A three-dimensional ultrasound prostate imaging system," Ultrasound in Medicine and Biology, vol. 22, pp. 735-746, 1996.
[85] R. W. Prager, A. Gee, and L. Berman, "Stradx: Real-time acquisition and visualisation of freehand 3D ultrasound," Medical Image Analysis, vol. 3, no. 2, pp. 129-140, 1999.
[86] D. Aiger and D. Cohen-Or, "Real-time ultrasound imaging simulation," Real-Time Imaging, vol. 4, no. 4, pp. 263-274, 1998.
[87] O. C. Zienkiewicz and R. L. Taylor, The Finite Element Method. Butterworth Heinemann, 2000.
[88] T. J. R. Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis. Mineola, New York: Dover Publications, 2000.
[89] M. Bro-Nielsen, "Finite element modeling in medical VR," Journal of the IEEE, vol. 86, no. 3, pp. 490-503, 1998.
[90] I. Wolfram Research, "Mathworld." http://mathworld.wolfram.com/WoodburyFormula.html.
[91] E. Keppel, "Approximating complex surfaces by triangulation of contour lines," IBM Journal of Research Developments, vol. 19, pp. 2-11, 1975.
[92] J.-D. Boissonnat, "Shape reconstruction from planar cross-sections," Computer Vision, Graphics and Image Processing, vol. 44, pp. 1-29, 1988.
[93] Institut National de Recherche en Informatique et en Automatique (INRIA), "Nuages," 2000-2004. http://www-sop.inria.fr/prisme/logiciel/nuages.html.
[94] E. J. Nicolson, Tactile sensing and control of a planar manipulator. PhD thesis, EECS, University of California Berkeley, 1987.
[95] S. Hang, TetGen, a quality tetrahedral mesh generator and three-dimensional delaunay triangulator, v1.3 user's manual. Weierstrass Institute for Applied Analysis and Stochastics (WIAS), Berlin, Germany, 9 ed., 2004. http://tetgen.berlios.de/.
[96] International Center for Numerical Methods in Engineering (CIMNE), "Gid: The personal pre and postprocessor," 2000-2004. http://gid.cimne.upc.es/.
[97] H.-W. Nienhuys and A. F. van der Stappen, "Interactive needle insertions in 3D nonlinear material," Tech. Rep. 2003-019, UU-CS: Utrecht University - Information and Computing Sciences, Utrecht, 2003.

\section*{Appendix A}

\section*{Finite Element Method}

\section*{A. 1 Definitions of Stress and Strain}

Consider a small area element \(\Delta A\), oriented with unit normal vector \(\hat{n}\), on the surface or within a deformable body under external forces as shown in Figure A.1. Let \(\Delta \boldsymbol{F}\) and \(\Delta \boldsymbol{M}\) be the resultant force and moment vectors, respectively, on \(\Delta A\). Their intensities on the area can be found at the limit as follows:
\[
\begin{align*}
& \lim _{\Delta A \rightarrow 0} \frac{\Delta \boldsymbol{F}}{\Delta A}=\frac{d \boldsymbol{F}}{d A}=\boldsymbol{T}  \tag{A.1}\\
& \lim _{\Delta A \rightarrow 0} \frac{\Delta M}{\Delta A}=\frac{d M}{d A}=\boldsymbol{C}
\end{align*}
\]
where \(\boldsymbol{T}\) is called stress vector or traction, and \(\boldsymbol{C}\) is called couple-stress vector. The elementary theory of elasticity proceeds on the assumption that \(\boldsymbol{C}=0\) [87]. \(\boldsymbol{T}\) is the stress intensity at a point with respect to a particular area element specified by the unit normal \(\hat{\boldsymbol{n}}\). A complete description of stress at a point requires the traction for all directions. This can be found by traction in principle directions.


Figure A.1: Resultant force and moment vectors on a small area element within a deformable body.

Consider an infinitesimal cube centered at a point inside a deformable body. Let the faces of the cube be oriented with the orthonormal basis vectors \(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\) of the coordinate frame as
shown in Figure A.2. Then the tractions on the faces with surface normals along the principle directions are shown as \(\boldsymbol{T}_{x}, \boldsymbol{T}_{y}, \boldsymbol{T}_{z}\). If they are written in terms of their Cartesian coordinates, e.g., \(\boldsymbol{T}_{x}=\sigma_{x x} \boldsymbol{x}+\sigma_{x y} \boldsymbol{y}+\sigma_{x z} \boldsymbol{z}\), the scalar stress components \(\sigma_{i j}\) form a second-order tensor \(\Psi\) defining the state of stress at a point as:
\[
\Psi=\left[\begin{array}{ccc}
\sigma_{x x} & \sigma_{x y} & \sigma_{x z}  \tag{A.2}\\
\sigma_{y x} & \sigma_{y y} & \sigma_{y z} \\
\sigma_{z x} & \sigma_{z y} & \sigma_{z z}
\end{array}\right]
\]
where the traction of an area element with unit normal \(\hat{\boldsymbol{n}}\) is given by \(\Psi^{T} \hat{\boldsymbol{n}}\).


Figure A.2: Components of stress on an infinitesimal cube.
Under the assumption that couple-stress does not exist, the equilibrium of zero moment implies \(\sigma_{i j}=\sigma j i\). Therefore the symmetric \(\Psi\) has only 6 independent variables, which can be expressed in the vector representation of stress as:
\[
\boldsymbol{\sigma}=\left[\begin{array}{cccccc}
\sigma_{x x} & \sigma_{y y} & \sigma_{z z} & \sigma_{y z} & \sigma_{z x} & \sigma_{x y} \tag{A.3}
\end{array}\right]^{T}
\]
where the first three components are called normal stresses, and the other three are called shear stresses.

Strain characterizes the deformation of a body under stress. Similar to the stress tensor \(\Psi\), a second-order strain tensor can completely describe the state of strain at a point:
\[
\Upsilon=\left[\begin{array}{ccc}
\epsilon_{x x} & \epsilon_{x y} & \epsilon_{x z}  \tag{A.4}\\
\epsilon_{y x} & \epsilon_{y y} & \epsilon_{y z} \\
\epsilon_{z x} & \epsilon_{z y} & \epsilon_{z z}
\end{array}\right]
\]

A more complicated mathematical treatment [87] shows the symmetry of \(\Upsilon\) for elastic bodies. Thus, \(\epsilon_{i j}=\epsilon_{j i}\), and the vector representation of strain becomes:
\[
\boldsymbol{\epsilon}=\left[\begin{array}{llllll}
\epsilon_{x x} & \epsilon_{y y} & \epsilon_{z z} & \epsilon_{y z} & \epsilon_{z x} & \epsilon_{x y} \tag{A.5}
\end{array}\right]^{T}
\]

\section*{A. 2 Deformation}

Displacements of a point \(\boldsymbol{x}=(x, y, z)\) can be written as \(\boldsymbol{u}=[u(\boldsymbol{x}) v(\boldsymbol{x}) w(\boldsymbol{x})]^{T}\) in the reference coordinate frame of the undeformed body. Then the strain components are given by:
\[
\begin{align*}
\epsilon_{x x} & =\frac{\partial u}{\partial x}+\frac{1}{2}\left[\left(\frac{\partial u}{\partial x}\right)^{2}+\left(\frac{\partial v}{\partial x}\right)^{2}+\left(\frac{\partial w}{\partial x}\right)^{2}\right]  \tag{A.6}\\
\epsilon_{x y} & =\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}+\left[\frac{\partial u}{\partial x} \frac{\partial u}{\partial y}+\frac{\partial v}{\partial x} \frac{\partial v}{\partial y}+\frac{\partial w}{\partial x} \frac{\partial w}{\partial y}\right] \tag{A.7}
\end{align*}
\]

The other four strain components are defined similarly. Note that the square-bracketed secondorder terms are relatively small for small displacements.

The matrix expression of the engineering strain tensor in terms of \(\boldsymbol{u}\) becomes:
\[
\begin{align*}
& \boldsymbol{\epsilon}(\boldsymbol{u})=\left[\begin{array}{c}
\epsilon_{x x} \\
\epsilon_{y y} \\
\epsilon_{z z} \\
\epsilon_{y z} \\
\epsilon_{z x} \\
\epsilon_{x y}
\end{array}\right]=\left[\begin{array}{c}
\frac{\partial u}{\partial x}+\frac{1}{2}\left[\left(\frac{\partial u}{\partial x}\right)^{2}+\left(\frac{\partial v}{\partial x}\right)^{2}+\left(\frac{\partial w}{\partial x}\right)^{2}\right] \\
\frac{\partial v}{\partial y}+\frac{1}{2}\left[\left(\frac{\partial u}{\partial y}\right)^{2}+\left(\frac{\partial v}{\partial y}\right)^{2}+\left(\frac{\partial w}{\partial y}\right)^{2}\right] \\
\frac{\partial w}{\partial z}+\frac{1}{2}\left[\left(\frac{\partial u}{\partial z}\right)^{2}+\left(\frac{\partial v}{\partial z}\right)^{2}+\left(\frac{\partial w}{\partial z}\right)^{2}\right] \\
\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}+\left[\frac{\partial u}{\partial y} \frac{\partial u}{\partial z}+\frac{\partial v}{\partial y} \frac{\partial v}{\partial z}+\frac{\partial w}{\partial y} \frac{\partial w}{\partial z}\right] \\
\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}+\left[\frac{\partial u}{\partial z} \frac{\partial u}{\partial x}+\frac{\partial v}{\partial z} \frac{\partial v}{\partial x}+\frac{\partial w}{\partial z} \frac{\partial w}{\partial x}\right] \\
\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}+\left[\frac{\partial u}{\partial x} \frac{\partial u}{\partial y}+\frac{\partial v}{\partial x} \frac{\partial v}{\partial y}+\frac{\partial w}{\partial x} \frac{\partial w}{\partial y}\right]
\end{array}\right] \\
& =\left[\begin{array}{ccc}
\frac{\partial}{\partial x} & 0 & 0 \\
0 & \frac{\partial}{\partial y} & 0 \\
0 & 0 & \frac{\partial}{\partial z} \\
0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\
\frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\
\frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0
\end{array}\right]\left[\begin{array}{l}
u \\
v \\
w
\end{array}\right]+\frac{1}{2}\left[\begin{array}{ccccccccc}
\frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} & \frac{\partial w}{\partial x} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{\partial u}{\partial y} & \frac{\partial v}{\partial y} & \frac{\partial w}{\partial y} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{\partial u}{\partial z} & \frac{\partial v}{\partial z} & \frac{\partial w}{\partial z} \\
0 & 0 & 0 & \frac{\partial u}{\partial z} & \frac{\partial v}{\partial z} & \frac{\partial w}{\partial z} & \frac{\partial u}{\partial y} & \frac{\partial v}{\partial y} & \frac{\partial w}{\partial y} \\
\frac{\partial u}{\partial z} & \frac{\partial v}{\partial z} & \frac{\partial w}{\partial z} & 0 & 0 & 0 & \frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} & \frac{\partial w}{\partial x} \\
\frac{\partial u}{\partial y} & \frac{\partial v}{\partial y} & \frac{\partial w}{\partial y} & \frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} & \frac{\partial w}{\partial x} & 0 & 0 & 0
\end{array}\right]\left[\begin{array}{l}
\frac{\partial u}{\partial x} \\
\frac{\partial v}{\partial x} \\
\frac{\partial w}{\partial x} \\
\frac{\partial u}{\partial y} \\
\frac{\partial v}{\partial y} \\
\frac{\partial w}{\partial y} \\
\frac{\partial u}{\partial z} \\
\frac{\partial v}{\partial z} \\
\frac{\partial w}{\partial z}
\end{array}\right] \\
& =B \boldsymbol{u}+\frac{1}{2} A \Theta \tag{A.8}
\end{align*}
\]

This vector \(\boldsymbol{\sigma}(\boldsymbol{u})\) is called the Green-Lagrange strain tensor. Neglecting the second-order
terms under small deformations leads to the linear strain approximation as follows:
\[
\begin{align*}
\boldsymbol{\sigma}(\boldsymbol{u}) & =\left[\begin{array}{ccc}
\frac{\partial}{\partial x} & 0 & 0 \\
0 & \frac{\partial}{\partial y} & 0 \\
0 & 0 & \frac{\partial}{\partial z} \\
0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\
\frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\
\frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0
\end{array}\right]\left[\begin{array}{c}
u \\
v \\
w
\end{array}\right] \\
& =B \boldsymbol{u} \tag{A.9}
\end{align*}
\]

This is called Cauchy's infinitesimal strain tensor. It is rotation variant, therefore must be used with care. This linear approximation will be used for the tissue model in this thesis since the tissue deformation and displacements during prostate brachytherapy is assumed to be small.

\section*{A. 3 Linear Elastic Behaviour}

Elastic materials load and unload along the same stress-strain curve. If they follow a linear path, they are called linear elastic materials. A Hookean elastic material obeys the generalized Hooke's law:
\[
\begin{equation*}
\sigma=C \epsilon \tag{A.10}
\end{equation*}
\]
where C is the material matrix relating the strain to the stress. For homogeneous and isotropic materials C is defined as:
\[
C=\left[\begin{array}{cccccc}
\lambda+2 \mu & \lambda & \lambda & 0 & 0 & 0  \tag{A.11}\\
\lambda & \lambda+2 \mu & \lambda & 0 & 0 & 0 \\
\lambda & \lambda & \lambda+2 \mu & 0 & 0 & 0 \\
0 & 0 & 0 & \mu & 0 & 0 \\
0 & 0 & 0 & 0 & \mu & 0 \\
0 & 0 & 0 & 0 & 0 & \mu
\end{array}\right]
\]
where the Lamé constants \(\lambda\) and \(\mu\) are expressed as follows:
\[
\begin{equation*}
\lambda=\frac{E \nu}{(1+\nu)(1-2 \nu)} \quad, \quad \mu=\frac{E}{2(1+\nu)} \tag{A.12}
\end{equation*}
\]

Here \(E\) and \(\nu\) are called Young's modulus and Poisson's ratio, respectively. The first one is the relation between the normal stress, \(\sigma_{x}\), and the normal strain, \(\epsilon_{x}\). The latter is the incompressibility of the material. In other words Poisson's ratio, \(\nu\), is the lateral contraction per longitudinal expansion.

A boundary-value problem of a linear elastic object \(\Omega\) consisting of particles \(\boldsymbol{x}=(x, y, z)\) can be defined as setting force or displacement constraints at some points and trying to solve for the displacements and forces of all other element particles. For the solution of such a problem, the strain energy of the elastic body is a convenient point to start from. It can be expressed as:
\[
\begin{equation*}
E_{\text {strain }}=\frac{1}{2} \int_{\Omega} \epsilon^{T} \sigma d \boldsymbol{x} \tag{A.13}
\end{equation*}
\]
where \(\epsilon\) is the engineering stress vector and \(\sigma\) is the engineering strain vector. Substituting the Hooke's law in (A.10):
\[
\begin{equation*}
E_{\text {strain }}=\frac{1}{2} \int_{\Omega} \epsilon^{T} C \epsilon d \boldsymbol{x} \tag{A.14}
\end{equation*}
\]

Then the strain energy stored in the body for the displacements \(\boldsymbol{u}(\boldsymbol{x})=\left[\begin{array}{lll}u & v & w\end{array}\right]^{T}\) can be written using \(\epsilon(\boldsymbol{u})=B \boldsymbol{u}\) from (A.9) as:
\[
\begin{equation*}
E_{\text {strain }}(\boldsymbol{u})=\frac{1}{2} \int_{\Omega}(B \boldsymbol{u}(\boldsymbol{x}))^{T} C(B \boldsymbol{u}(\boldsymbol{x})) d \boldsymbol{x} \tag{A.15}
\end{equation*}
\]

Assuming the potential energy of an object at rest is \(E(0)\), its strain energy and the work done on the body particles by the external/internal forces, \(f(x)\), can be related using the conservation of energy as follows:
\[
\begin{equation*}
E(\boldsymbol{u})=\frac{1}{2} \int_{\Omega}(B \boldsymbol{u})^{T} C(B \boldsymbol{u}) d \boldsymbol{x}-\frac{1}{2} \int_{\Omega} \boldsymbol{f}^{T} \boldsymbol{u} d \boldsymbol{x} \tag{A.16}
\end{equation*}
\]

Note that \((B \boldsymbol{u})^{T} \neq \boldsymbol{u}^{T} B^{T}\) since \(B\) is an operator matrix.

\section*{A. 4 Discretization}

Since an analytical continuous solution of (A.10) over a complex shape is computationally impossible, one possible approximation can be the discretization of the object into simpler geometrical structures. The behaviour of the body, \(\Omega\), can be modeled by dividing it into the elements \(\Omega^{i}\) where \(i=1, \ldots, s\) as in Figure A. 3 having the nodes \(P_{q}\) where \(q=1, \ldots, n\). Although those elements ought not necessarily have the same structure, assume that they are all tetrahedrons in our implementation. Considering an element, \(\Omega^{e}\), shown in Figure A. 4 its corners are expressed as \(P_{i}^{e}\) where \(i\) is the local node number within the element and it is completely unrelated to the global node numbering.

While the \(i^{t h}\) node displacement is \(\boldsymbol{u}_{i}=\left[\begin{array}{lll}u_{i} & v_{i} & w_{i}\end{array}\right]^{T}\), the compound displacement vector can be written as \(\boldsymbol{u}=\left[\begin{array}{llll}\boldsymbol{u}_{1}^{T} & \boldsymbol{u}_{2}^{T} \ldots \boldsymbol{u}_{n}^{T}\end{array}\right]^{T}\). For the tetrahedron element, \(\Omega^{e}\), the compound element displacement vector becomes \(\boldsymbol{u}^{e}=\left[\left(\boldsymbol{u}_{1}^{e}\right)^{T}\left(\boldsymbol{u}_{2}^{e}\right)^{T}\left(\boldsymbol{u}_{3}^{e}\right)^{T}\left(\boldsymbol{u}_{4}^{e}\right)^{T}\right]^{T}\).

The displacement of a point inside an element can be estimated as a weighted average of its corner displacements:
\[
\begin{equation*}
\boldsymbol{u}(\boldsymbol{x})=\sum_{i=1}^{4} N_{i}^{e}(\boldsymbol{x}) \mathbf{u}_{i}^{e} \tag{A.17}
\end{equation*}
\]


Figure A.3: A 2-D discretization example of a body \(\Omega\) into finite elements \(\Omega^{i}\).


Figure A.4: A tetrahedral element and its corners in local numbering.
where \(N_{i}^{e}\) are called the basis functions. A linear displacement interpolation satisfies zero-order continuity between elements. Although it was reported that a quadratic interpolation gives much better FEA results, for our purposes the accuracy is given up for the speed and the simplicity of the linear estimation.

The linearly interpolating basis functions are called natural coordinates. Natural coordinates should be a function of the position, \(\boldsymbol{x}\), since the weighting of displacements should depend on where the internal node is. For example, if the node is closer to a corner, then this corner's displacement will determine the node's displacement the most. And also the total displacement shouldn't be magnified which brings the constraint \(\sum_{i} N_{i}^{e}=1\). Consider the relocation of the corners onto the origin which requires negative displacements of their 3D coordinates. This would confine any internal point to the origin as well, resulting in a displacement equal to the negative of its coordinate. Using this fact, given the element corner positions \(\left[x_{i} y_{i} z_{i}\right]\), the position of an internal point \([x y z]\) can be estimated by the natural coordinates as follows:
\[
\left[\begin{array}{l}
1  \tag{A.18}\\
x \\
y \\
z
\end{array}\right]=\left[\begin{array}{cccc}
1 & 1 & 1 & 1 \\
x_{1} & x_{2} & x_{3} & x_{4} \\
y_{1} & y_{2} & y_{3} & y_{4} \\
z_{1} & z_{2} & z_{3} & z_{4}
\end{array}\right]\left[\begin{array}{c}
N_{1}^{e}(\boldsymbol{x}) \\
N_{2}^{e}(\boldsymbol{x}) \\
N_{3}^{e}(\boldsymbol{x}) \\
N_{4}^{e}(\boldsymbol{x})
\end{array}\right] \Rightarrow\left[\begin{array}{c}
N_{1}^{e}(\boldsymbol{x}) \\
N_{2}^{e}(\boldsymbol{x}) \\
N_{3}^{e}(\boldsymbol{x}) \\
N_{4}^{e}(\boldsymbol{x})
\end{array}\right]=\left[\begin{array}{cccc}
a_{1}^{e} & b_{1}^{e} & c_{1}^{e} & d_{1}^{e} \\
a_{2}^{e} & b_{2}^{e} & c_{2}^{e} & d_{2}^{e} \\
a_{3}^{e} & b_{3}^{e} & c_{3}^{e} & d_{3}^{e} \\
a_{4}^{e} & b_{4}^{e} & c_{4}^{e} & d_{4}^{e}
\end{array}\right]\left[\begin{array}{l}
1 \\
x \\
y \\
z
\end{array}\right]
\]
where:
\[
\left[\begin{array}{llll}
a_{1}^{e} & b_{1}^{e} & c_{1}^{e} & d_{1}^{e}  \tag{A.19}\\
a_{2}^{e} & b_{2}^{e} & c_{2}^{e} & d_{2}^{e} \\
a_{3}^{e} & b_{3}^{e} & c_{3}^{e} & d_{3}^{e} \\
a_{4}^{e} & b_{4}^{e} & c_{4}^{e} & d_{4}^{e}
\end{array}\right]=\left[\begin{array}{cccc}
1 & 1 & 1 & 1 \\
x_{1} & x_{2} & x_{3} & x_{4} \\
y_{1} & y_{2} & y_{3} & y_{4} \\
z_{1} & z_{2} & z_{3} & z_{4}
\end{array}\right]^{-1}
\]

So the linearly interpolating basis functions for a tetrahedron is expressed as:
\[
\begin{equation*}
N_{i}^{e}(\boldsymbol{x})=a_{i}^{e}+b_{i}^{e} x+c_{i}^{e} y+d_{i}^{e} z \tag{A.20}
\end{equation*}
\]

To calculate \(B \boldsymbol{u}(\boldsymbol{x})\) in (A.16) inside an element \(e\), the partial derivatives of the discretized displacements in (A.17) are needed:
\[
\begin{align*}
\frac{\partial \boldsymbol{u}}{\partial x} & =\sum_{i=1}^{4} \frac{N_{i}^{e}(\boldsymbol{x})}{\partial x} \mathbf{u}_{i}^{e}=\sum_{i=1}^{4} b_{i}^{e} u_{i}^{e} \\
\frac{\partial \boldsymbol{u}}{\partial y} & =\sum_{i=1}^{4} \frac{N_{i}^{e}(\boldsymbol{x})}{\partial y} \mathbf{u}_{i}^{e}=\sum_{i=1}^{4} c_{i}^{e} u_{i}^{e}  \tag{A.21}\\
\frac{\partial \boldsymbol{u}}{\partial z} & =\sum_{i=1}^{4} \frac{N_{i}^{e}(\boldsymbol{x})}{\partial z} \mathbf{u}_{i}^{e}=\sum_{i=1}^{4} d_{i}^{e} u_{i}^{e}
\end{align*}
\]

And similarly \(\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}}\) and \(\frac{\partial \boldsymbol{w}}{\partial \boldsymbol{x}}\) are obtained:
\[
\begin{align*}
\frac{\partial \boldsymbol{v}}{\partial x} & =\sum_{i=1}^{4} b_{i}^{e} v_{i}^{e} & \frac{\partial \boldsymbol{w}}{\partial x} & =\sum_{i=1}^{4} b_{i}^{e} w_{i}^{e} \\
\frac{\partial \boldsymbol{v}}{\partial y} & =\sum_{i=1}^{4} c_{i}^{e} v_{i}^{e} & \frac{\partial \boldsymbol{w}}{\partial y} & =\sum_{i=1}^{4} c_{i}^{e} w_{i}^{e}  \tag{A.22}\\
\frac{\partial \boldsymbol{v}}{\partial z} & =\sum_{i=1}^{4} d_{i}^{e} v_{i}^{e} & \frac{\partial \boldsymbol{w}}{\partial z} & =\sum_{i=1}^{4} d_{i}^{e} w_{i}^{e}
\end{align*}
\]

In (A.21) and (A.22) all the partial derivatives are constants. Therefore this constant \(B \boldsymbol{u}(\boldsymbol{x})\) can be rewritten inside an element as:
\[
B \boldsymbol{u}(\boldsymbol{x})=\sum_{i=1}^{4} B_{i}^{e} u_{i}^{e} \quad \text { where } B_{i}^{e}=\left[\begin{array}{ccc}
b_{i}^{e} & 0 & 0  \tag{A.23}\\
0 & c_{i}^{e} & 0 \\
0 & 0 & d_{i}^{e} \\
0 & d_{i}^{e} & c_{i}^{e} \\
d_{i}^{e} & 0 & b_{i}^{e} \\
c_{i}^{e} & b_{i}^{e} & 0
\end{array}\right]
\]

If this is expressed in one matrix notation using the \(12 \times 1\) compound element displacement vector, \(\boldsymbol{u}^{e}, B \boldsymbol{u}(\boldsymbol{x})\) becomes \(B^{e} \boldsymbol{u}^{e}\) for any \(\boldsymbol{x}\) inside a given element \(e\) where:
\[
B^{e}=\left[\begin{array}{cccccccccccc}
b_{1}^{e} & 0 & 0 & b_{2}^{e} & 0 & 0 & b_{3}^{e} & 0 & 0 & b_{4}^{e} & 0 & 0  \tag{A.24}\\
0 & c_{1}^{e} & 0 & 0 & c_{2}^{e} & 0 & 0 & c_{3}^{e} & 0 & 0 & c_{4}^{e} & 0 \\
0 & 0 & d_{1}^{e} & 0 & 0 & d_{2}^{e} & 0 & 0 & d_{3}^{e} & 0 & 0 & d_{4}^{e} \\
0 & d_{1}^{e} & c_{1}^{e} & 0 & d_{2}^{e} & c_{2}^{e} & 0 & d_{3}^{e} & c_{3}^{e} & 0 & d_{4}^{e} & c_{4}^{e} \\
d_{1}^{e} & 0 & b_{1}^{e} & d_{2}^{e} & 0 & b_{2}^{e} & d_{3}^{e} & 0 & b_{3}^{e} & d_{4}^{e} & 0 & b_{4}^{e} \\
c_{1}^{e} & b_{1}^{e} & 0 & c_{2}^{e} & b_{2}^{e} & 0 & c_{3}^{e} & b_{3}^{e} & 0 & c_{4}^{e} & b_{4}^{e} & 0
\end{array}\right]
\]

Now the strain energy in (A.15) can be rewritten as a sum of element energies using the property of definite integrals, \(\int_{V}() d V=\sum_{e} \int_{V^{e}}() d V\), as:
\[
\begin{equation*}
E_{\text {strain }}(\boldsymbol{u})=\frac{1}{2} \sum_{e} \int_{\Omega^{e}}\left(B^{e} \boldsymbol{u}^{e}\right)^{T} C\left(B^{e} \boldsymbol{u}^{e}\right) d \boldsymbol{x}=\frac{1}{2} \sum_{e} \int_{\Omega^{e}} \boldsymbol{u}^{e T} B^{e T} C B^{e} \boldsymbol{u}^{e} d \boldsymbol{x} \tag{A.25}
\end{equation*}
\]

Note that whereas \(B\) is an operator matrix, \(B^{e}\) is a constant matrix. That's how the equality \(\left(B^{e} \boldsymbol{u}^{e}\right)^{T}=\boldsymbol{u}^{e T} B^{e T}\) holds. So the strain energy of an element can be expressed as:
\[
\begin{align*}
E_{\text {strain }}^{e}\left(\boldsymbol{u}^{e}\right) & =\frac{1}{2} \int_{\Omega^{e}} \boldsymbol{u}^{e T} B^{e T} C B^{e} \boldsymbol{u}^{e} d \boldsymbol{x} \\
& =\frac{1}{2} \boldsymbol{u}^{e T} B^{e T} C B^{e} \boldsymbol{u}^{e} \int_{\Omega^{e}} d \boldsymbol{x} \\
& =\frac{1}{2} \boldsymbol{u}^{e T} B^{e T} C B^{e} \boldsymbol{u}^{e} v^{e} \tag{A.26}
\end{align*}
\]
where \(v^{e}\) is the element volume.

\section*{A. 5 Element energy}

Considering the tendency towards the minimum potential energy, (A.16), in the tissue, the solution of the strain problem occurs when the variation of the energy is zero, \(\delta E(\boldsymbol{u})=0\). Then the discretized form becomes:
\[
\begin{equation*}
\delta E(\boldsymbol{u})=\sum_{e} \delta E^{e}\left(\boldsymbol{u}^{e}\right)=0 \tag{A.27}
\end{equation*}
\]

Considering \(\sum_{e} \delta E^{e}\left(\boldsymbol{u}^{e}\right)=\sum_{e} \frac{\partial E^{e}}{\partial \boldsymbol{u}^{e}} \delta \boldsymbol{u}^{e}\) and each node displacement, \(\boldsymbol{u}_{i}^{e}\), is independent from the others, the following is concluded:
\[
\begin{equation*}
\forall_{i \in \Omega^{e}}: \frac{\partial E^{e}}{\partial \boldsymbol{u}_{i}^{e}}=0 \tag{A.28}
\end{equation*}
\]

As the element energy definition, \(E^{e}\), the equation (A.16) can be written using the element
strain energy expression in (A.26) as follows:
\[
\begin{equation*}
E^{e}\left(\boldsymbol{u}^{e}\right)=\frac{1}{2} \boldsymbol{u}^{e T} B^{e T} C B^{e} \boldsymbol{u}^{e} v^{e}-\boldsymbol{f}^{e} \tag{A.29}
\end{equation*}
\]
where \(\boldsymbol{f}^{e}\) is the \(12 \times 1\) compound element force vector. Applying vanishing criteria in (A.28) on this expression using the algebraic fact that \(\frac{\partial}{\partial \boldsymbol{x}} \boldsymbol{x}^{T} \boldsymbol{A} \boldsymbol{x}=2 \boldsymbol{A} \boldsymbol{x}\) gives:
\[
\begin{align*}
\frac{\partial}{\partial \boldsymbol{u}^{e}}\left(\frac{1}{2} \boldsymbol{u}^{e T} B^{e T} C B^{e} \boldsymbol{u}^{e} v^{e}-\boldsymbol{f}^{e}\right) & =0 \\
\frac{v^{e}}{2} \frac{\partial}{\partial \boldsymbol{u}^{e}}\left(\boldsymbol{u}^{e T} B^{e T} C B^{e} \boldsymbol{u}^{e}\right)-\boldsymbol{f}^{e} & =0 \\
v^{e} B^{e T} C B^{e} \boldsymbol{u}^{e}-\boldsymbol{f}^{e} & =0 \\
K^{e} u^{e} & =\boldsymbol{f}^{e} \tag{A.30}
\end{align*}
\]
where \(K^{e}\) is the \(12 \times 12\) element stiffness matrix. Calculating these for each element and then combining them to form the entire set of system stiffness equations yields:
\[
\begin{equation*}
\boldsymbol{K} \boldsymbol{u}=f \tag{A.31}
\end{equation*}
\]
where \(\boldsymbol{K}\) is the global stiffness matrix, \(\boldsymbol{u}\) is the compound node displacements, and \(\boldsymbol{f}=\) \(\left[\begin{array}{llll}f_{1}^{T} & f_{2}^{T} & \ldots & f_{n}^{T}\end{array}\right]^{T}\) is the \(3 n \times 1\) compound force vector such that \(f_{i}=\left[\begin{array}{lll}f_{i}^{x} & f_{i}^{y} & f_{i}^{z}\end{array}\right]^{T}\) is the force components at the node \(i\). Combining the set of system equations of each element necessitates the mapping of the local element node numbering to the global node numbers:
\[
\begin{align*}
\boldsymbol{K} & =\sum_{e} \operatorname{global}\left(K^{e}\right)  \tag{A.32}\\
\boldsymbol{f} & =\sum_{e} \operatorname{global}\left(f^{e}\right) \\
\boldsymbol{u} & =\operatorname{global}\left(u^{e}\right)
\end{align*}
\]

Note that the displacements are not summed not to accumulate the displacement of the same node shared by the neighbouring elements, whereas their force contributions and thus the individual stiffness matrices are summed up.

Since \(\boldsymbol{K}\) is singular, the linear system in (A.31) cannot be solved. One way of solving is
modifying \(\boldsymbol{K}\) so that the displacements of some nodes are constrained as follows:
\[
\begin{gather*}
{\left[\begin{array}{cccccc}
k_{11} & k_{12} & \cdots & k_{1 i} & \cdots & k_{1 n} \\
k_{21} & k_{22} & \cdots & k_{2 i} & \cdots & k_{2 n} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
k_{i 1} & k_{i 2} & \cdots & k_{i i} & \cdots & k_{i n} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
k_{n 1} & k_{n 2} & \cdots & k_{n i} & \cdots & k_{n n}
\end{array}\right] \quad \boldsymbol{u}=\left[\begin{array}{c}
f \\
f_{1} \\
f_{2} \\
\vdots \\
f_{i} \\
k_{11}
\end{array} k_{12}\right.} \\
k_{21}  \tag{A.33}\\
k_{22}
\end{gather*} \cdots
\]

Here setting \(d_{i}=0\) fixes the node to its initial position. The necessity of this node constraining process can also be visualized intuitively: A body floating in space without any constraints can move to anywhere even with no force exerted. This would result in infinitely many solutions of the system. It can easily be seen that fixing 3 non-collinear nodes in space, constrains the body (both translations and rotations). Although 3 nodes bring 9 displacement constraints, it can analytically be proven that six independent constraints are enough to ensure that \(\boldsymbol{K}\) is invertible.

\section*{Appendix B}

\section*{Boundary Condition Change}
(2.6) gives us a set of equations which we can write the row \(i\) and the rest seperately as:
\[
\begin{align*}
f_{i} & =k_{i 1} u_{1}+\ldots+k_{i i} u_{i}+\ldots+k_{i r} u_{r}  \tag{B.1}\\
\forall_{j \in \aleph^{i} i}: f_{j} & =k_{j 1} u_{1}+\ldots+k_{j i} u_{i}+\ldots+k_{j r} u_{r} \tag{B.2}
\end{align*}
\]
where \(\aleph^{i}=\{1, \ldots, i-1, i+1, \ldots, r\}\).
The form in (2.7) requires \(f_{j}\) and \(u_{i}\) in terms of \(u_{j}\) and \(f_{i}\). If we find \(u_{i}\) from (B.1) and substitute it in (B.2) it becomes \({ }^{1}\) :
\[
\begin{align*}
\Rightarrow u_{i}= & -\frac{1}{k_{i i}}\left[k_{i 1} u_{1}+\ldots+0^{i i}+\ldots+k_{i r} u_{r}-f_{i}\right]  \tag{B.3}\\
\forall_{j \in \aleph^{i}}: \quad f_{j}= & k_{j 1} u_{1}+\ldots+0^{j i}+\ldots+k_{j r} u_{r} \\
& -\frac{k_{j i}}{k_{i i}}\left[k_{i 1} u_{1}+\ldots+0^{i i}+\ldots+k_{i r} u_{r}-f_{i}\right]
\end{align*}
\]

To be able to write it as a difference from the initial \(K^{-1}\) matrix like in (2.8) we add and subtract the term \(k_{j i} f_{i}\) and reorder the terms:
\[
\begin{align*}
\forall_{j \in \aleph^{i}}: \quad f_{j}= & k_{j 1} u_{1}+\ldots+k_{j i} f_{i}+\ldots+k_{j r} u_{r} \\
& -\frac{k_{j i}}{k_{i i}}\left[k_{i 1} u_{1}+\ldots+\left(k_{i i}-1\right) f_{i}+\ldots+k_{i r} u_{r}\right] \tag{B.4}
\end{align*}
\]

To make the \(u_{i}\) in (B.3) also look like the form in (2.7) we should add and subtract all the

\footnotetext{
\({ }^{1}\) The terms \(0^{i i}\) and \(0^{j i}\) are used to indicate the absence of the terms with the factors \(k_{i i}\) and \(k_{j i}\), respectively.
}
missing terms and regroup them properly:
\[
\begin{align*}
\Rightarrow u_{i}= & k_{i 1} u_{1}+\ldots+k_{i i} f_{i}+\ldots+k_{i r} u_{r} \\
& -\left[k_{i 1} u_{1}+\ldots+k_{i i} f_{i}+\ldots+k_{i r} u_{r}\right] \\
& -\frac{1}{k_{i i}}\left[k_{i 1} u_{1}+\ldots+0^{i i}+\ldots+k_{i r} u_{r}\right]+\frac{1}{k_{i i}} f_{i} \\
\Rightarrow u_{i}= & k_{i 1} u_{1}+\ldots+k_{i i} f_{i}+\ldots+k_{i r} u_{r} \\
& -\frac{k_{i i}+1}{k_{i i}}\left[k_{i 1} u_{1}+\ldots+0^{i i}+\ldots+k_{i r} u_{r}\right]+\frac{1-k_{i i}{ }^{2}}{k_{i i}} f_{i} \\
\Rightarrow u_{i}= & k_{i 1} u_{1}+\ldots+k_{i i} f_{i}+\ldots+k_{i r} u_{r} \\
& -\frac{k_{i i}+1}{k_{i i}}\left[k_{i 1} u_{1}+\ldots+\left(k_{i i}-1\right) f_{i}+\ldots+k_{i r} u_{r}\right] \tag{B.5}
\end{align*}
\]

In (B.4) and (B.5) the upper line terms are the initial coefficients of the \(K^{-1}\) matrix. We can rewrite the common square bracketed terms in matrix form as:
\[
k_{i 1} u_{1}+\ldots+\left(k_{i i}-1\right) f_{i}+\ldots+k_{i r} u_{r}=\left[\begin{array}{lllll}
k_{i 1} & \cdots & \left(k_{i i}-1\right) & \cdots & k_{i r}
\end{array}\right]\left[\begin{array}{c}
u_{1}  \tag{B.6}\\
\vdots \\
f_{i} \\
\vdots \\
u_{r}
\end{array}\right]
\]

And the coefficients of these bracketed terms become:
\[
\left[\begin{array}{c}
-\frac{k_{1 i}}{k_{i i}}  \tag{B.7}\\
\vdots \\
-\frac{k_{i i}+1}{k_{i i}} \\
\vdots \\
-\frac{k_{r i}}{k_{i i}}
\end{array}\right]=-\frac{1}{k_{i i}}\left[\begin{array}{c}
k_{1 i} \\
\vdots \\
k_{i i}+1 \\
\vdots \\
k_{r i}
\end{array}\right]
\]```


[^0]:    ${ }^{1}$ Gleason score is a cancer aggressiveness grade determined via biopsy.

[^1]:    ${ }^{2}$ Haptic comes from a Greek term for touch. Haptics is the science concerned with the tactile or kinesthetic sense of touch.

[^2]:    ${ }^{3}$ This data set is available on the internet.

[^3]:    ${ }^{4}$ It is available on the internet at [66]

[^4]:    ${ }^{1}$ Matlab ${ }^{\circledR}$ is a registered trademark of The MathWorks, Inc.

[^5]:    ${ }^{1}$ Its free academic version [96] allows up to 700 nodes with 3000 tetrahedra.

[^6]:    ${ }^{1}$ This figure does not depict the mesh. The lines on it are only for representing the TRUS planes and the general structure of the tissue.

[^7]:    ${ }^{1}$ The needle geometry is rotationally symmetric. The tip bevel disobeys this axial symmetry, however, the bevel is not a concern of FEA either. Thus, $\theta$ is taken constant which saves many operations during transformations.

