Elasticity Parameter Estimation in a Simple Measurement Setup

by

Motahareh Tekieh

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.A.Sc. degree in
Biomedical Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Elastic deformation has wide applications in medical simulations, therefore when it comes to designing physical behavior of objects for realistic training applications, determining material parameters so that objects behave in a desired way becomes a crucial. In this work we consider the problem of elasticity parameter estimation for deformable bodies, which is important for accurate medical simulations.

Our work has two major steps: the first step is the data acquisition and preparation, and the second step is the parameter estimation. The experimental setup for data acquisition consists of depth and force sensors. Surface deformations are acquired as a series of images along with the corresponding applied forces. The contact point of the force sensor on the surface is found visually and the corresponding applied forces are acquired directly from the force sensor. A complete mesh deformation which is obtained from a surface tracking method is employed along with force measurements in the elasticity parameter estimation method.

Our approach to estimate the physical material properties is based on an inverse linear finite element method. We have experimented with two approaches to optimize the elasticity parameters: a linear iterative method and a force-displacement error minimization method. The two methods were tested on the simulation data, and the second method was tested on three deformable objects. The results are presented for the data captured by two different depth sensors. The result is a set of two parameters, the Young's modulus and the Poisson's ratio, which represents the stiffness of the object under test.

Acknowledgements

First and for most, I would like to thank my supervisor Professor Jochen Lang for his generous support and kind advices throughout my study. It has been a great pleasure having the opportunity to work under his supervision.

Additionally, I would like to thank Stefanie Wuhrer for her cooperation and helpful advices.

Last but not least, I would like to thank my parents, and my husband, Reza, for their warm love, support and encouragement.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years, there has been an increasing trend in medical (training) simulators to draw on physics-based modeling from computer graphics. For modeling the properties of soft tissues, the material parameters are employed to simulate their visual and tactile behavior in a realistic way. In this thesis we study the elasticity parameter estimation of deformable solids. In Section 1.1, background about physics-based modeling along with the motivation is presented. Sections 1.2 to 1.4 present the thesis statement, the overview of the approach, and the main contributions of the thesis respectively. Finally, the order in which this thesis is organized is stated in Section 1.5.

1.1 Physical Behavior of Objects

In medical training applications and surgery simulations [49, 13, 60, 66], physics-based deformation plays a major role in simulating realistic interactions. The simulations aim to improve the skills of the medical team and allow them to perform pre-operative planning. As minimally invasive surgery develops, there is a need to train physicians in many novel procedures in different pathologies. One of the key problems in the development of a medical simulator is that the physical parameters of anatomical structures must be defined [17]. Although it has a special importance in medical applications, other application domains can also benefit from defining

physical properties for interaction objects. For instance, robotic applications where robots interact with real world objects need to know the physical parameters of the interaction objects. An example of this is in domestic applications [24] where robots interact with both soft and hard objects like clothes, plants, furniture, etc. They need to know the physical parameters in order to apply an appropriate amount of force both to be effective and non-destructive, since extraneous force might destroy fragile objects. The prediction of the appropriate amount of force is only possible when the physical properties of the object are modeled and simulated. Tuning the parameters manually in order to obtain a plausible look can be applied to computer games and movies, but does not necessarily result in a physically realistic computation of the interaction forces in the virtual environment [24]. It should be noted that even the most detailed deformation modeling will be unrealistic if the simulated model does not involve physical properties. Therefore, when it comes to designing physical behavior of objects for realistic training applications, determining material parameters so that objects behave in a realistic manner becomes a problem, which is the main purpose of this thesis.

Several researchers have investigated the concept of physical behavior of an object and defined related properties. The physical behavior of an object is defined by a large number of its physical properties, but in this thesis, the focus is on elasticity parameters of soft solids, which are the main concern in solid deformation. In linear elasticity there are two parameters that describe the stiffness and deformability of an isotropic homogeneous linear elastic solid. These are known as Young's Modulus E and Poisson's ratio ν, which describe elasticity and compressibility of the material, respectively [7]. These two parameters known as elastic moduli are one pair from the elastic moduli set that are all related to each other. In other words, the elastic moduli are convertible parameters that any two of them can define the material properties. The elastic moduli are used to construct the stiffness matrix $\boldsymbol{K}$ in Hooke's law, where external force $\boldsymbol{f}$ exerted on the object is related to the resulting displacement $\boldsymbol{u}$ (Equation 1.1).

$$\boldsymbol{f} = \boldsymbol{K}\boldsymbol{u} \tag{1.1}$$

Another pair of elasticity parameters or elastic moduli is called Lamé parameters namely the Lamé's first parameter λ and the shear modulus μ. The Lamé parameters are used in this thesis to simplify the linear estimation of the stiffness matrix.

Estimation of elasticity parameters of an object is feasible when sufficient information about the physical phenomena is available through an estimation framework. This framework consists of a measurement system, a computational model for the object, and a computational approach. The input data for the framework can be acquired in different system setup, as long as the setup is able to measure the contact position of the object, the contact force, and the displacement of the interaction surface. In our estimation framework, the measurement system comprises a force sensor with an attached checkerboard as visual target for tracking and a depth sensor (either a stereo camera or Kinect sensor). The model of the object is assumed to be a homogeneous linear isotropic elastostatic body. The computational approach consists of 3 main phases: data acquisition and preprocessing, surface tracking, and parameter estimation. The data acquisition includes range scanning of an object and the force recording whereas the preprocessing involve background subtraction to clean the acquired point cloud and tracking of the checkerboard target. The second phase is the surface tracking of the object which is handled with the method by Wuhrer et al. [75]. The third and final phase is making use of all data prepared to estimate the parameters of the object in the linear estimation approach.

Determining the parameters of a deformable model based on contact forces and the behavior of the object is known as an inverse problem. Since, this concept is of mechanical solid and structure interest [36], the measurement is mostly done in mechanical testing laboratories [32]. Even though, in mechanical labs the computational methods are more precise, the verification of the methods is usually performed by comparing it to analytical solution or finer discretization of the model domain [39]. In our work, once we compare the results obtained from the data captured by two depth sensors for different objects. The other time, we compare the results of two estimation methods for one object.

The biomedical applications of our estimation method is divided into two main parts which can be used independently. The first part are methods to use a simple, portable measurement setup to obtain force measurements (Section 4.1.3) and pose of the probe (Section 4.2.3). The capture time is the time needed to record a video. Our methods can be employed in remote medicine applications [52] [16] where an examination is performed by a non-expert and the captured files including a video and a data file for the force can be transferred to the remote station. The second part is our estimation method (Section 3.3.4) which is tolerant to high noise scenarios. It can be used to estimate the elasticity parameters of any organ by feeding in the input data consists of displacement and force. It can be even employed to estimate the elasticity of internal organs for which we have the force and displacement information via ultrasound [20]. Currently, the run time of our estimation method makes this an off-line process.

1.2 Thesis statement

Physically-based simulations require the knowledge of the material parameters of the designed objects so that they behave in a realistic way. In this regard, a method is required to estimate the elastic parameters of a real world deformable object based on its recorded point cloud data and force measurements, while being deformed by an external force.

1.3 Thesis overview

Figure 1.1 shows an overview of the system's pipeline, from data capture to final results. The capture stage produces rectified color images and scene point clouds for the visual part, and force magnitude and direction measurements on the tactile part. The majority of the system falls within the pre-processing and parameter estimation blocks, which comprises several stages. For pre-processing, the first stage is performing a background subtraction algorithm on the color images in order to find a mask for foreground point clouds of the scene. A second stage is the visual tracking of the checkerboard target in the scene followed by calculating the transformation of force and position vectors to the camera frame. In the surface tracking stage,

a closed triangle surface mesh of the object known as template mesh is deformed and aligned with the captured frames. The surface tracking which we contributed to was mainly developed and implemented by Wuhrer [75], and will be explained as part of our system setup. Finally, having the displacements on the template mesh and the corresponding force vector, a pair of mechanical parameters is estimated.

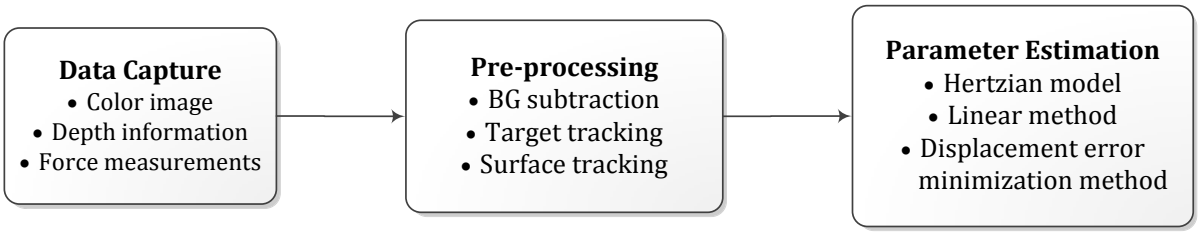


Figure 1.1: An overview of the system.

1.4 Contribution

The following outlines the contributions of our work:

- Designing a method for acquiring data and an algorithm to process the data for the parameter estimation. The data is acquired using portable devices that include a single depth sensor along with a force sensor. The measurement setup has the advantage of being simple, it does not involve any expensive or heavyweight devices like a CT scanner or MRI.

- We combine a data acquisition system with a surface tracking method to estimate elasticity parameters of real world objects, which is a step toward simplifying parameter estimation methods.

- Our contribution to the tracking step by Wuhrer et al. [75] is the processing of the captured data, the joint development of tracking methods in line with our estimation method, and testing of the method in our capture framework.

1.5 Thesis organization

The rest of this thesis is organized as follows:

- Chapter 2 reviews in detail the related work introduced above, providing background on methods used for parameter estimation and data acquisition.
- Chapter 3 explains the theory of continuum mechanics and the Finite Element Method as far as required for this thesis, followed by two approaches for parameter estimation: linear and displacement error minimization. Each of these methods are tested with data obtained from linear FEM simulation.
- Chapter 4 gives an overview of the acquisition process for our system, including the hardware and software components and its calibration. It also includes the preprocessing steps needed to prepare the data for the parameter estimation step.
- Chapter 5 provides, compares and discusses the results based on data captured from three objects. These results include the parameter estimations obtained from Hertzian contact model and displacement error minimization approach.
- Chapter 6 summarizes and concludes the work and proposes future extensions.

The mathematical notation employed throughout this thesis are as follows: capital-bold letters such as $\boldsymbol{K}$ denote matrices, small-bold letters such as $\boldsymbol{u}$ denote vectors, normal font letters such as E denote scalars, and blackboard-bold letters such as $\mathbb{F}$ denote vertex sets.

Chapter 2

Related Work

In this chapter we discuss related work on data acquisition and elasticity parameter estimation of deformable objects. Section 2.1 introduces the acquisition systems employed in order to measure the deformation of objects. Surface tracking techniques are presented in Section 2.2. Physically based models of deformable objects are then discussed in Section 2.3, followed by material models in Section 2.4. Elasticity parameter estimation methods are presented in Section 2.5. Finally this chapter is summarized in Section 2.6.

2.1 Data Acquisition

To estimate the mechanical parameters of deformable object researchers have used two main approaches. The first approach is using simulation data with known forces and displacements. Although most researchers used simulation data to test their method, some eventually tested their method with real data. Section 2.1.2 presents the devices employed in different studies.

2.1.1 Simulation data

In the literature there are some works that tested their method solely on simulation data. Becker and Teschner [3] presented an approach for the estimation of elasticity parameters employing the finite element method based on simulation data. To demonstrate the method is

robust enough for real data, they have tested their method with added noise to simulated data. Frank et al. [23] computes the deformation cost against travel cost in a virtual environment with deformable objects in order to direct a mobile robot. Lee and Lin [42] estimates the linear elasticity parameters of deformable 2D shapes by minimizing the distance between simulated model and target model. In this thesis we have used simulation data of a foam block, to test the parameter estimation method.

2.1.2 Force and video frame capture

Estimating the elasticity parameters based on real world data is known as data driven approaches and requires a certain amount of infrastructure. Many systems were proposed or employed in the literature to measure the force applied to the object and record the resulting displacement during the deformation procedure. The experimental setup used by different researchers can be classified based on the testing objects: either they are designed for a specific applications like soft tissue or cloth, or they can be used for multi-purpose applications.

Some of the devices were specifically used to examine a particular object.

Chen et al. [14] used an Instron Universal Material Testing Machine, which are motorized computer-driven devices to apply force and measure the resulting displacement via ultrasound. Testing machine can handle different tests consisting of tensile, compression, shear, flexure, peel, tear, cyclic and bend tests. However, they are pretty big (from 60 cm to more than one meter high) and are limited to relatively small samples. Another limitation of Chen et al.'s setup [14] is that the displacements are measured in 1D using ultrasound.

Benech and Negreira [4] placed the test object between a central frequency ultrasonic transducer and a computer-driven vibrator. This technique is only capable of measuring micro displacements.

Schwartz et al. [66] used a load cell attached to a stepper motor for performing a compression experiment on liver tissue.

Hall et al. [27] designed a less expensive system based on a balance placed under a gelatin sample. While a motorized computer-driven compressor deforms the sample, the applied force is measured using the balance. The limitation for this device is that the maximum compression range used in the measurement is 5 mm and the direction of the measured deformation is also limited to one axis.

Miller and Chinzei [51] captured the deformation on brain tissue using a tensile stress machine with high precision. However, the machine is big and relatively slow, with maximum compression speed of 0.64 mm per second.

Wang et al. [73] used a setup similar to tensile testing for stretching cloth in 3 directions. The cloth is stretched by hanging known weights and the amount of stretch is measured using the labeled points. Miguel et al. [50] used force sensor and stereo camera to have a complete 3D information of the cloth deformation procedure.

Wang and Hirai [74] captured square blocks of sweets being compressed using a camera and a force sensor.

Other experimental setups are more multi-purpose and are not designed for a specific application. Lang et al. [40] used a Puma 260 robot arm, a six-axis force/torque sensor along with a trinocular stereo vision system integrated on a tele-robotic active measurement facility called ACME [59]. The force sensor acquires the applied force while deforming the object and the stereo camera captures the deformation. The surface displacement is obtained using the stereo data and the optical flow. This device can acquire a precise set of measurements from objects of medium sizes. However, ACME is large build around an optical bench and some of its components are expensive.

Liao et al. [43] used a motion capture system with 2 cameras as their optical tracking system to find the nodal displacement based on active LED markers on the object's surface, along

with a pressure sensor recording the external force exerted on the object, which is calibrated by a back propagation neural network. In contrast to our setup where one camera (either the reference camera from the Bumblebee stereo camera or from the Kinect) is employed and no active surface tracking is involved.

Closest to the acquisition method discussed in this thesis is the acquisition system in the work performed by Frank et al. [24] on estimating the elasticity parameters. They optimize the elasticity parameters by establishing the relation between applied forces and corresponding surface deformations that was acquired using a 7 DOF robot manipulator equipped with a force-torque sensor and a stereo Bumblebee camera. However, the approach discussed in this thesis is distinct in that it captures the deformation free hand and no robotic device is involved.

Syllebranque and Boivin [69] presented a force capture system which is a combination of a force sensor and a tracking device (phantom haptic device). The acquisition system comprises a handy cam, a video projector, a force sensor mounted on a phantom arm. An ARTookKit pattern is also used for positioning the real world object in the simulated scene, similar to our checkerboard for positioning the force sensor in the real world.

Bickel et al. [7] used a capture system that consists of force probes and a trinocular stereo system that occupies a large space. The three cameras are synchronized using an external trigger system. The surface displacements are measured using the markers painted on the object.

Boonvisut et al. [8] captured the deformation of the silicon rubber phantoms with a stereo camera and record force measurements through an ATI Nano17 6-axis force-torque sensor. The force sensor is mounted on a robotic gripper which is able to grasp one side of the object and pull it.

Ahn and Kim [1] performed soft tissue loading using an optical 3D deformation analyzer to obtain surface deformation and one dimensional indentation device to measure the force. The device is restricted to one dimensional loading.

2.2 Surface tracking

One of the requirements for elasticity parameter estimation is a displacement of the soft object during the deformation. The methods that employ only simulation data, obtain the exact displacement via simulation [3]. However, the experimental approaches need a method to track the surface deformation and compute the global displacement. These methods may be divided into two main categories: methods that employ dedicated hardware or physical means, and methods that used computer vision techniques.

Liao et al. [43] obtained the nodal displacement based on active LED markers on the object's surface. Syllebranque and Boivin [69] presented a capture system which is a combination of a force sensor and a tracking device (phantom haptic device) that tracks the surface deformation. Wang and Hirai [74] determined the amount of compression of sweet blocks by comparing the measured points on the object surface against the background measurements. Boonvisut et al. [8] tracked a soft tissue phantom deformation during the manipulation by some markers placed on its surface. Ahn and Kim [1] tracked the surface of porcine liver in a loading test by coloring it in black and white sprinkles to create visual patterns. The stereo images were recorded from original and deformed patterns. Then the position, displacement, and plane strain tensor of the tissue surface are calculated using a photogrammetric evaluation procedure.

Frank et al. [24] used a depth camera to capture the scene and an error minimization approach is performed iteratively in order to minimize the difference between the object under deformation and the simulation. Wang et al. [73] tracked the cloth deformation by minimizing the difference between simulated feature points and observed feature points

Lang and Pai [40] estimated the surface displacement using a range-flow calculation technique. To calculate the range-flow, simultaneous 2D optical flow is combined with stereo depth information. For each vertex location in view of the camera, displacement is measured visually. The approach takes advantage of redundant images provided by a trinocular stereo camera to increase the robustness of the range flow calculation.

2.3 Deformable models

Physically-based simulations require a deformable model to represent the simulated object and its behavior. Two popular methods used to model the deformable solids include: the particle or mass-spring systems, and the continuum mechanics models. Nealen et al. [56] surveyed deformable models in Computer Graphics. In this section we study the deformable models employed to represent the deformable solid for parameter estimation.

2.3.1 Particle system (Mass-spring system)

The simplest deformable model used in simulations are mass-spring systems or particle systems. Particle systems are discrete structures consisting of point masses connected together by a network of massless springs. These models can handle large deformations and are used very often in real-time simulations. However they cannot represent physical properties of the material due to lack of a systematic relationship between spring constants. Another issue with particle systems is their inconsistent behavior as the mesh resolution and topology varies [56].

Several researchers tried to estimate the spring constants using learning algorithms and compared the resulting simulations with a reference model like the video of real object deformation [5] [12], or the corresponding Finite Element Model [6] [45] to evaluate the estimated parameter. The disadvantages of these methods are that they are computationally expensive, and require a reference model, as well as that the result of a specific method may not hold for different mesh resolutions.

2.3.2 Volumetric model

In the continuum mechanics method, the deformable object is represented by a volumetric model, which is a closer model to the real world objects, defined by material parameters which determines the behavior under applied forces. There are a variety of mathematical solution methods that are used in continuum mechanics including the Finite Element Method (FEM), the Finite Difference Method (FDM), the Finite Volume Method (FVM), and the Boundary Element Method (BEM). All of these techniques are used to solve numerically the Partial Differential Equations (PDE) of the physical rules applicable to elastic solids.

The FEM is the most popular in computer graphics and simulation as can be seen from its more frequent use in the literature compared to the other three methods. In the Finite Element Method the elastic solid is assumed as a continuous volume that is discretized into a finite number of elements. Thus, the continuous function applicable to the domain as a whole is approximated by the sum of discrete functions valid for each element. In other words, the problem is reduced to approximated PDEs for each element. An advantage of the Finite Element Method over particle systems is that the material properties can be integrated directly into FE model. However, it is computationally expensive.

Several researchers have applied linear FEM (see Section 3.2 for more detail) for surgery simulations, amongst which is the work of Bro-Nielsen and Cotin [11]. In order to speed up the computations, they also adopted a condensed model to simulate only the surface nodes, similar to BEM. Similar to our work, Liao et al. [43] have used linear FEM to estimate the elasticity parameters for soft tissues. Furthermore, this method is employed for cloth parameter estimation [73], or it has been reduced to 2D FEM to model 2D shape deformation [42] [74].

Linear FEM uses linearized Green strain relation known as the Cauchy's strain relation to represent the deformation behavior of the object, which is only valid for small deformations (see Section 3.1.1). Since the Cauchy's strain tensor is not independent of rigid body rotations,

in the case of large rotational deformations the object grows abnormally due to ghost forces. To eliminate these artifacts, Müller et al. [53] proposed a method called co-rotational FEM, that separates the rigid rotation and deformation in the linear Cauchy' strain relation. In this method, the nodal deformation forces are first computed in the current configuration aligned with initial configuration, then rotated back to the current configuration.

$$\boldsymbol{f_i} = \boldsymbol{R}\sum_{j=1}^{n}\boldsymbol{K_{ij}}\left(\boldsymbol{R}^T\boldsymbol{p_j} - \boldsymbol{p_{0j}}\right) \tag{2.1}$$

Equation 2.1 shows the deformation force computation for the nodes, where i and j are element node indices, $\boldsymbol{R}$ is the rotation matrix between initial and current configuration, $\boldsymbol{K_{ij}}$ is the corresponding element stiffness matrix, $\boldsymbol{p}$ are the positions of element nodes in current configuration, $\boldsymbol{p_0}$ are the positions in initial configuration, and $\boldsymbol{p} = \boldsymbol{p_0} + \boldsymbol{u}$. In order to estimate the local rotation at a node between two configurations, orthonormal basis vectors are defined such that they are based on the direction of adjacent edges, then the transformation between them are calculated. The transformation matrix contains the rotational part, and the translational part is omitted. In this approach the forces are computed with respect to the non-rotated reference frame. This yields stable and visually pleasing results. The researchers that employed co-rotational FEM to support large deformations for the purpose of parameter estimation include Frank et al. [24], Syllebranque et al. [69], and Bickel et al. [7].

The main difference of FEM with other numerical methods is the domain discretization. In FDM the object is sampled using a regular mesh grid which is easier to implement. Terzopoulos et al. [71] were the leading team to apply FDM to physically based animations. In FVM, the finite volume is the small box that surrounds the mesh node in the discretized domain. Teran et al. [70] applied FVM in their skeletal muscle simulation. The major disadvantage of the FDM and FVM is that they are not appropriate for irregular geometries. Since we use objects with different geometry, none of these two models are appropriate to use. Unlike the previous methods that perform the computations on the volume of the deformable body, in BEM the

surface or boundary of the model is discretized. The problem with the BEM is that the topological changes like cuts of the object are difficult to handle. Lang et al. [39] used BEM to represent their deformable model for elasticity parameter estimation.

2.4 Constitutive material models

The constitutive law in an elastic model defines the relationship between the strain and the stress. The majority of studies in the literature uses a linear elastic material model due to its simplicity in implementation and low computational cost. However, for some applications, mostly soft tissue modeling, non-linear elasticity models are employed too. This section explains the material model used by researchers.

The linear material model (see Section 3.1.3 for more detail) which is derived from Hook's law (Equation 1.1) is used by several researchers in order to estimate the two elasticity parameters. They employed linear FEM for solving the problem numerically [3] [43] [42] [39]. However, the linear FEM does not support large deformations due to some artifacts. Thus, an alternative mathematical method called corotational FEM [53] is used by other researchers that require to model large deformations with the linear material model [24] [69].

The non-linear material models are also used by a number of researchers for soft tissue parameter estimation. A non-linear material model that is used more often is the Neo-Hookean model in which the relationship between strain and stress becomes non-linear after being linear at the beginning. An advantage of this model is that it still uses the material parameters (Lamé parameters λ and μ) that can be interpreted physically. The strain energy density function W that relates the Cauchy-Green deformation tensor $\boldsymbol{C}$ to the strain energy density is shown in Equation 2.2. This is the characteristic function for stress-strain relationship. Equation 2.3 shows the stress tensor for Neo-Hookean material model.

$$W = \frac{1}{2}\left(\mu\left(i_1 - 3\right) - \mu\log\left(i_3\right) + \lambda\left(\left(\sqrt{i_3} - 1\right)^2\right)\right) \tag{2.2}$$

$$\boldsymbol{S} = \mu \boldsymbol{I} - \mu \boldsymbol{C}^{-1} + \lambda \left(\sqrt{i_3} - 1\right) \sqrt{i_3} \boldsymbol{C}^{-1} \tag{2.3}$$

i_1, i_2, and i_3 are the invariants of the Cauchy-Green deformation tensor and are calculated as

$$i_1 = trace\left(\boldsymbol{C}\right)$$
$$i_2 = \tfrac{1}{2}\left(trace\left(\boldsymbol{C}\right)^2 - trace\left(\boldsymbol{C}^2\right)\right)$$
$$i_3 = \det\left(\boldsymbol{C}\right)$$

Boonvisut et al. [8] employed a Neo-Hookean material model to model non-linearity of soft tissues, while experimenting with silicon rubber phantoms. Lee et al. [41] used both linear and non-linear (Neo-Hookean) material model to estimate Young's modulus of soft tissue based on CT images.

2.5 Estimation methods

Elastic parameter estimation is an active field of research both in mechanical engineering and in Computer Graphics. In mechanical engineering, elasticity parameters are obtained by a well-known test called tensile test, in which a uniaxial loading is performed on the test object [28]. A suitable strain energy function is then derived based on comparison of the experimental data curves with analytical stress-strain relations. Finally the set of material parameters are estimated accordingly [48] [64].

In Computer Graphics similar tests are also performed to retrieve the elasticity parameters, but with different devices and methods. The majority of the experimental approaches in the literature try to estimate the elasticity parameters by applying a known force to a discretized model and measure the resulting displacement. The elasticity parameters can then be optimized knowing these information. This approach is employed for a variety of applications which can be classified into elastography, soft tissue simulations, cloth simulations, and multi-purpose applications.

In the multi-purpose application category, Becker and Teschner [3] used a constraint optimization to solve for the physically meaningful Young's Modulus and the Poisson's ratio from simulated data. They presented a linear structure of the stiffness matrix that reduces the optimization problem to linear least square. The advantages of this work are that the result is independent of the shape or discretization of the object and due to its linear nature, the optimization is not trapped in local minima. The linear method in this thesis is based on this work. Lang et al. [37] presented an iterative method to solve for Poisson's ratio and shear modulus which used a discretized Green's function matrix for linear elastic deformation. The test objects used are an anatomical soft wrist model and a plush toy. Syllebranque et al. [69] presented a method to estimate the Poisson's ratio solely based on the video of the solid deformation. The Young's modulus is then estimated using the measured forces. The results showed that the Young's modulus and Poisson's ratio can be computed separately for an object and are not correlated. Bickel et al. [7] presented a method to model heterogeneous and non-linear material objects. However, each measured sample of the object is modeled as a linear material using the corotational FEM. An extra data point is added to each tetrahedron to represent the overall non-linear behavior of the object. Finally, all the parameters are interpolated to demonstrate a nonlinear material. Frank et al. [24] optimized the elasticity parameters in a realistic simulation while minimizing the distance between observed and simulated deformed surfaces. They used the Iterative Closest Point (ICP) algorithm for registering the two surfaces, and a gradient descent algorithm to minimize the error function. Wang and Hirai [74] presented a specific non-linear material model for Japanese sweets consisting of two elastic and three viscous parameters. They minimized the distance between the measured and simulated data in 3 steps: displacement minimization in the first and the last step and force minimization in the second step, to retrieve the material parameters. Cretu et al. [19] presented a method based on neural network to model the elastic characteristics of deformable objects.

The soft tissue parameter estimation is mainly used for surgery simulations [10] [57]. Kauer et al. [34] estimated the biological soft tissue (human uteri) using a non-linear material model.

They optimized the parameters by fitting the experimental data with a 2D FEM using Levenberg-Marquardt algorithm. Choi and Zheng [15] estimated the Young's modulus and the Poisson's ratio of a sample soft tissue (articular cartilage) using two different size indenters. Cowan et al. [18] used recorded MRI images with different applied pressures to the heart muscle tissue to measure the displacement of deformation and performed a non-linear optimization to model the sample with Neo-Hookean material model. Soza et al. [68] also estimated the brain tissue parameters based on MRI data. They used an iterative method that combines the finite element simulation with an image registration based on MR data. Boonvisut et al. [8] presented a method to retrieve the parameters of rubber phantoms based on data obtained from surgical robot interacting with the object. They optimized the difference between both observed and simulated force and observed and simulated displacement. This is a similar error function we used in the displacement error minimization section of this thesis. Lister et al. [44] estimated the parameters of a porcine liver by minimizing the difference between the experimental force and the simulated force.

The other concept in soft tissue parameter estimation domain is elastography [61] which extracts the elasticity parameters of soft tissues for the purpose of medical diagnosis. Usually the Young's modulus or viscoelastic properties of suspicious tissues are stiffer and have different characteristics compared to normal soft tissues. Doyley [20] provides an elaborate survey about elastography including the employed material models and problem solving techniques. The most common medical imaging used to produce elastograms is ultrasonic imaging [58] [62]. In this technique two ultrasound images are taken, one at rest state, and the other one after applying a known force to the skin. Based on these images, the deformation field is estimated. The other medical imaging devices are also used for this purpose including MRI [63] [68] and CT [41].

A broad research domain is about cloth property analysis, estimation and simulation. Due to vast use of the cloth simulation in physically-based animations, many researchers tried data-driven methods to model variety of cloth types and optimized the corresponding parameters

by fitting the experimental data with simulation data in different configurations like planar, bending, and stretching using both linear [73] and non-linear [50] material model.

Some studies reduced one dimension and used a 2D FEM to simplify the problem. Schnur and Zabaras [65] employed a 2D FEM to solve for Young's modulus for a given Poisson's ratio. They optimized the corresponding error function which was in the form of non-linear least square, using the Levenberg-Marquardt method. Lee and Lin [42] presented a method to estimate the elasticity parameters by minimizing the difference between simulated and given target displacements. In order to speed up the optimization they used a technique to reduce a dimension on the displacement vector using a set of orthonormal bases. The reduced bases technique increases the optimization performance by reducing the accuracy.

2.6 Summary

In this chapter, we discussed the related work regarding different aspects of this thesis. Different devices and infrastructures are used for collecting required data for elasticity parameter estimation and physically-based simulations, including massive robotic devices, medical imaging devices, and simpler devices. However, they all have to be capable of measuring the contact forces and resulting deformations. In order to model the test object mainly the linear material model is used in the literature both supporting large deformations using corotational FEM or limited to infinitesimal displacements using linear FEM. For applications that require more accurate modeling like soft tissue modeling, non-linear materials are also employed. Finally, an objective function is employed in order to minimize the difference between observed and simulated data. The majority of the optimizations are based on displacement differences, but some are also based on forces.

Chapter 3

Elasticity Parameter Estimation

This chapter focuses on the theory part of the linear elasticity and describes the two approaches applied to retrieve the elasticity parameters. In Section 3.1 the continuum model for elasticity and the related mechanical concepts are explained. Section 3.2 introduces the Finite Element Method for the solution of linear elastic objects. Section 3.3 describes linear and displacement error minimization method to estimate the object's elasticity parameters. The methods are then tested on data from a simulation. Finally, Section 3.4 presents a summary of the chapter.

3.1 Continuum Model

Continuum mechanics is the study of material behavior as a continuous mass rather than discrete particles. Different material behavior is described by different continuum models such as elasticity, plasticity, and viscosity. This thesis focuses on linear elasticity which is a common model for elastic material. Linear elasticity is usually used to simplify the model used for simulation and estimation. Hook's law is an accurate approximation of linear elasticity, in which under small forces and deformations, displacement is linearly proportioned to the applied force. Hook's law for a linear spring is

$$\boldsymbol{f} = \boldsymbol{K}\boldsymbol{u} \tag{3.1}$$

where $\boldsymbol{f}$ is the applied force, $\boldsymbol{K}$ is the spring constant, $\boldsymbol{u}$ is the spring elongation or compression. A general form of Hook's law models the behavior of material, taking into account some internal quantities. This behavior is described by the relationship between four physical quantities: displacement, strain, stress, and force. Figure 3.1 illustrates the relation between these external and internal mechanical quantities. As shown in the the figure, displacement or movement of particles in the continuum object is related to rate of dimension change or strain by kinematic laws. Strain is related to the internal force measure or stress by constitutive laws of material. Finally, stress is related to external force by equilibrium equations. While force and displacement are external quantities that can be observed and measured, stress and strain are internal mathematical factors for measuring the effect of force and displacement. External quantities can be manipulated by force and displacement boundary conditions. The relationship between stress and strain determines the physical behavior of continuum object. In the general form of Hook's law, strain of the material is proportional to stress, each of which have independent components. Therefore, the factor that relates these two quantities can no longer be a real number like the spring constant mentioned earlier, but rather a matrix of real numbers that represents a linear map. The theory description for elastic solids in this chapter is based on the theses of Bro-Neilsen[9] and of Jerabkova [31]. To read more about continuum mechanics, see [47].

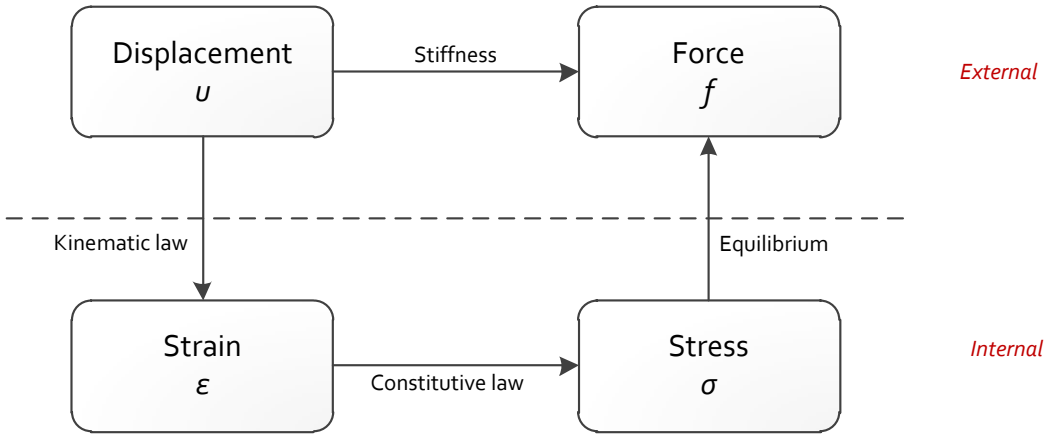


Figure 3.1: The relation between external and internal mechanical quantities.

3.1.1 Strain

Strain measures the rate of displacement change in deforming a solid. For a one dimensional spring, strain is the ratio of elongation with respect to original length, but for a 3D object, strain is defined as a tensor

$$\boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_{xx} & \epsilon_{xy} & \epsilon_{xz} \\ \epsilon_{yx} & \epsilon_{yy} & \epsilon_{yz} \\ \epsilon_{zx} & \epsilon_{zy} & \epsilon_{zz} \end{bmatrix} \tag{3.2}$$

where

$$\epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} + \sum_{k=1}^{3} \frac{\partial u_k}{\partial x_i}.\frac{\partial u_k}{\partial x_j}\right) \tag{3.3}$$

Equation 3.2 is called the Green strain tensor where u is the displacement vector, x is the coordinate of a point, and the indices i, j, k represent the 3 coordinates in 3D space. Since the strain tensor is symmetric, it can be written as a strain vector $\boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_{xx} & \epsilon_{yy} & \epsilon_{zz} & \epsilon_{xy} & \epsilon_{xz} & \epsilon_{yz} \end{bmatrix}^T$. The strain tensor is invariant under rigid body motion (translation and rotation) and thus independent of the deformation path. Usually, under large deformations when the displacement gradient is also large, the nonlinear Equation 3.3 is used. For small deformations the nonlinear term is approximated to zero and the equation is simplified to the linear Cauchy tensor.

$$\epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{3.4}$$

However, it should be noted that the nonlinear term is large when the rotational component is significant, so the linearized equation cannot be used. Neither can it be used for plastic deformations that are dependent on the deformation path. The nonlinearity of Equation 3.3 which is in the displacement-strain relation is called geometrical nonlinearity, while strain-stress nonlinearity is known as material nonlinearity. In linear elasticity, both relations are linearized.

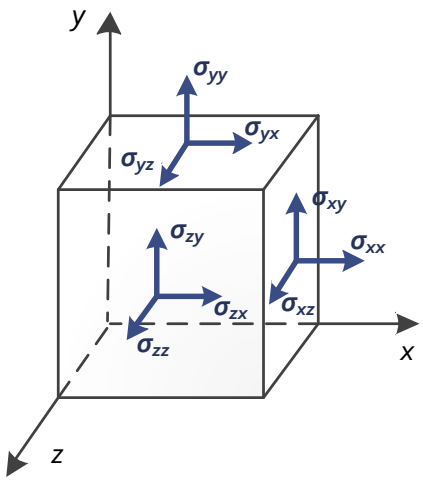


Figure 3.2: Stress components on an infinitesimal cube inside the object.

3.1.2 Stress

Stress measures the internal force inside the solid that balances the external and body forces applied to the solid. A stress vector is a traction vector which is force per unit area. Consider Figure 3.2 that defines a set of perpendicular planes to Cartesian axes at an arbitrary point inside the object. Stress on this point can be expressed as three traction vectors on these planes, each of them comprises three orthogonal components. Overall nine components express the stress on these three planes which are organized into the stress tensor.

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \tag{3.5}$$

where σ_{ij} is the stress on i-th plane along j-th direction. As shown in the Figure 3.2, diagonal components which are normal to surfaces represent normal stress, and off-diagonal components which are tangential to the surfaces represent shear stress. Normal stress changes the volume of the object while shear stress deforms the object without volume change. Similar to the strain tensor, the stress tensor is symmetric and can be written in vector notation

$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} & \sigma_{yy} & \sigma_{zz} & \sigma_{xy} & \sigma_{xz} & \sigma_{yz} \end{bmatrix}^T$. The stress and strain tensors are related by the constitutive material law which is explained in Section 3.1.3. In addition, the equilibrium equations for the infinitesimal interior cube can be written as Equation 3.6.

$$\boldsymbol{f_i} = \sum_{j=1}^{3} \frac{\partial \sigma_{ji}}{\partial x_j} \tag{3.6}$$

3.1.3 Constitutive Law

The constitutive law relates stress to strain depending on material type. For linear elastic materials, the constitutive law is the generalization of Hook's law

$$\sigma_{ij} = \sum_{k=1}^{3} \sum_{l=1}^{3} (C_{ijkl} \epsilon_{kl}) \tag{3.7}$$

where C_{ijkl} is a fourth order tensor of elastic constants with $3^4 = 81$ components. However, an anisotropic material has only 21 independent components, and an isotropic material has only 2 constants. These are Lamé constants λ and μ or alternatively Young's modulus $E = \mu(2\mu + 3\lambda)/(\mu + \lambda)$ and Poisson's ratio $\nu = \lambda/(2(\mu + \lambda))$.

Young's modulus is the constant that represents the ratio of applied tensile stress and the resulting strain.

$$E = \frac{\sigma_{tensile}}{\epsilon_{tensile}} \tag{3.8}$$

In linear elastic material, loading an object axially in xx direction causes a lateral contraction in yy and zz directions. The ratio between axial and lateral displacement is known as Poisson's ratio which is between 0 and 0.5.

$$\nu = -\frac{\epsilon_{lateral}}{\epsilon_{axial}} \tag{3.9}$$

For isotropic materials, this ratio is

$$\nu = -\frac{\epsilon_{yy}}{\epsilon_{xx}} = -\frac{\epsilon_{zz}}{\epsilon_{xx}} \tag{3.10}$$

The Lamé constants are used to write the constitutive relation more concisely. The constitutive law of Equation 3.7 can be written as Equation 3.11, where $\boldsymbol{C}$ is the material matrix known as rigidity or constitutive matrix.

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{yz} \\ \tau_{zx} \\ \tau_{xy} \end{bmatrix} = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix} \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \gamma_{yz} \\ \gamma_{zx} \\ \gamma_{xy} \end{bmatrix} = \boldsymbol{C\epsilon} \tag{3.11}$$

$$\gamma_{ij} = \epsilon_{ij} + \epsilon_{ji} = 2\epsilon_{ij}$$

$$\epsilon_i = \epsilon_{ii}$$

3.2 Finite Element Method

The Finite Element Method is a numerical technique that solves for boundary values by minimizing an error function. The main idea of FEM is to approximate the solution for a domain by solving the problem for geometrically discretized sub domains known as finite elements. The continuum mechanics laws are applied to the elements and a set of simultaneous equations are obtained. Linear algebra or non-linear methods are then employed to solve the equations in order to find the unknown boundary values. One of the advantages of FEM is that the accuracy of the result can be improved by increasing the resolution of the model. In this section an overview of the FEM is presented. To read more about FEM, see [2], [77], and other related references.

3.2.1 Principle of virtual work

Virtual work is referred to a work that is a result of either virtual force and a real displacement or real force and virtual displacement (any arbitrary possible displacement by the particle). According to the principle of virtual work, for a body in equilibrium the work done by internal stresses is canceled out and is only limited to work done by the applied external forces (Equation 3.12). The major benefit of this principle is that only forces that do work during the virtual displacement are considered in the mechanics of the system.

$$\tilde{\boldsymbol{W}}_{\boldsymbol{I}} = \tilde{\boldsymbol{W}}_{\boldsymbol{E}} \tag{3.12}$$

On the other hand, the external virtual work done by force $\boldsymbol{f}$ acting on node with virtual displacement $\tilde{\boldsymbol{u}}$ is Equation 3.13, and the internal virtual work associated with this force is Equation 3.14 where $\tilde{\boldsymbol{\epsilon}}$ is the virtual strain caused by virtual displacement $\tilde{\boldsymbol{u}}$.

$$\tilde{\boldsymbol{W}}_{\boldsymbol{E}} = \tilde{\boldsymbol{u}}^T \boldsymbol{f} \tag{3.13}$$

$$\tilde{\boldsymbol{W}}_{\boldsymbol{I}} = \int_v \tilde{\boldsymbol{\epsilon}}^T \boldsymbol{\sigma} dV \tag{3.14}$$

Having the strain-displacement relationship expressed as $\boldsymbol{\epsilon} = \boldsymbol{B}\boldsymbol{u}$, it can be substituted in Equation 3.14 along with Equation 3.11. As a result of internal and external virtual work equality, it yields

$$\tilde{\boldsymbol{u}}^T \boldsymbol{f} = \tilde{\boldsymbol{u}}^T \left(\int_v \boldsymbol{B}^T \boldsymbol{C} \boldsymbol{B} dV \right) \boldsymbol{u} \tag{3.15}$$

so the governing equation of linear elasticity is obtained as follows with $\left(\int_v B^T C B dV\right)$ representing the stiffness.

$$\boldsymbol{f} = \left(\int_v \boldsymbol{B}^T \boldsymbol{C} \boldsymbol{B} dV \right) \boldsymbol{u} \tag{3.16}$$

3.2.2 The Discretization

In FEM, the domain is discretized into finite elements such as tetrahedra and triangles, and the displacement and force are approximated over these elements. Thus, force and displacement are only evaluated at nodes, so any external force also needs to be expressed as a nodal force with the same effect. The force and displacement values within an element are interpolated by nodal values using what is known as shape functions. In other words, the shape function ϕ_i interpolates the nodal displacement $\boldsymbol{u_i}$ to get a continuous displacement $u(\boldsymbol{x})$ for any arbitrary point within an element.

$$u(\boldsymbol{x}) = \sum_{i=1}^{n} \phi_i(\boldsymbol{x}) \boldsymbol{u_i} \tag{3.17}$$

The shape function has this feature of being 1 at node i and 0 at other nodes, which results in $u(\boldsymbol{x_i}) = \boldsymbol{u_i}$. Therefore, it is the same as barycentric coordinates for linear triangles and tetrahedra.

Equation 3.16 that satisfies the continuity condition, can be written in discretized form with nodal force vectors and displacement vectors.

$$\boldsymbol{f_i} = \sum_{j=1}^{n} \int_v \boldsymbol{B_i}^T \boldsymbol{C} \boldsymbol{B_j} dV \boldsymbol{u_i} = \sum_{j=1}^{n} \boldsymbol{K_{ij}} \boldsymbol{u_j} \tag{3.18}$$

$$\boldsymbol{K_e} = V_e \boldsymbol{B}^T \boldsymbol{C} \boldsymbol{B} \tag{3.19}$$

In Equation 3.19, matrix $\boldsymbol{K_e}$ is the stiffness matrix of the element, which is symmetric and has the size of $[3n \times 3n]$, where n is the number of element nodes. Since, linear tetrahedra are employed in this thesis as finite elements, the stiffness matrix for an element is of size $[12 \times 12]$. V_e represents the tetrahedron volume.

3.2.3 FEM Assembly

In order to model the behavior of the object as a whole, all the elements together have to be considered. The assembly of elements should be in such a way that first, the displacement of all nodes meet at the same node must be the same, and second, the sum over all interior element forces at a particular node must be equal to external force at that node. It should be noted that the interior element forces exerted on a node are associated only with elements contributing that node. Thus the global stiffness matrix is assembled simply by adding the element stiffness matrices contributing to a particular node, while considering the global numbering of the nodes. The resulting matrix is a symmetric, $[3n \times 3n]$ matrix, where n is the number of mesh nodes. The global stiffness matrix is a sparse matrix that depends on the connection between elements.

3.3 Parameter Estimation

In this section, two approaches to retrieve the elasticity parameters are presented based on the linear Finite Element Model described in the previous section. A discretized model of a block is assumed with the base boundary nodes fixed and the rest of the nodes unconstrained. A force vector is applied to a triangle on the top of the model. The force causes displacement on the unconstrained points. Knowing the forces and displacements of nodes, the elasticity parameters are estimated such that the stiffness matrix minimizes the following relation.

$$\min_{\lambda,\mu} \|\boldsymbol{K}\boldsymbol{u} - \boldsymbol{f}\|_2^2 \tag{3.20}$$

In order to minimize equation 3.20, the relation is arranged such that it is linear in terms of the unknown elastic parameters. This is similar to Becker and Teschner's approach [3]. First, the relation for a single tetrahedron is explained. It is then, generalized to a tetrahedral mesh.

For a single tetrahedron, matrix $\boldsymbol{C}$ in Equation 3.11 can be decomposed, so that elasticity parameters are scalars of two matrix terms.

$$\boldsymbol{C} = \lambda \boldsymbol{C_1} + \mu \boldsymbol{C_2} = \lambda \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \mu \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.21}$$

Thus, the stiffness matrix can be written as

$$\boldsymbol{K_e} = V_e \boldsymbol{B}^T \left(\lambda \boldsymbol{C_1} + \mu \boldsymbol{C_2}\right) \boldsymbol{B} = \lambda \boldsymbol{K_{e1}} + \mu \boldsymbol{K_{e2}} \tag{3.22}$$

and the Equation 3.20 is expressed as

$$\min_{\lambda,\mu} \|\lambda \boldsymbol{K_{e1}} \boldsymbol{u} + \mu \boldsymbol{K_{e2}} \boldsymbol{u} - \boldsymbol{f}\|_2^2 \tag{3.23}$$

where $\boldsymbol{K_e}$ depends only on the undeformed geometry. As Equation 3.23 shows, the optimization problem is turned into linear least square problem. In general, matrices $\boldsymbol{K_{e1}}$ and $\boldsymbol{K_{e2}}$ are assembled into the global matrices $\boldsymbol{K_1}$ and $\boldsymbol{K_2}$, respectively, for the complete tetrahedral mesh. Since these are decomposed stiffness matrices, the assembly is also done in the same way as for the stiffness matrix. This will result in the global stiffness matrix as

$$\boldsymbol{K} = \lambda \boldsymbol{K_1} + \mu \boldsymbol{K_2} \tag{3.24}$$

and the optimization equation as

$$\min_{\lambda,\mu} \|\lambda \boldsymbol{K_1} \boldsymbol{U} + \mu \boldsymbol{K_2} \boldsymbol{U} - \boldsymbol{F}\|_2^2 \tag{3.25}$$

where $\boldsymbol{U}$ and $\boldsymbol{F}$ are the global displacement and force vectors, respectively. The equation can be written in the form of linear least squares.

$$\min_{\lambda,\mu} \left\| \begin{bmatrix} \boldsymbol{K_1 U} & \boldsymbol{K_2 U} \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \end{bmatrix} - \boldsymbol{F} \right\|_2^2 \tag{3.26}$$

3.3.1 Nodal Force and Displacement calculation

In this thesis a contact triangle is considered as the contact region. The tip of the force sensor which is the actual contact point is located using the result of visual tracking. The closest triangle to the contact point is selected as the contact triangle. This means the measured force is exerted on the vertices of this triangle. Figure 3.3 shows the contact area on the surface of the mesh, with the red triangle being the contact triangle. Since the indenter mounted on the force sensor used in our experiments is a sphere, the actual contact area on the object's surface can be approximated a circle. The traction on the actual contact area which is shown by a red dashed circle should be equal to the traction on the contact region of the mesh, which is shown by a green shape in the figure. Equation 3.27 shows that the nodal force $\boldsymbol{f_{node}}$ is calculated by knowing the measured force $\boldsymbol{f_{measured}}$, the actual contact area $A_{indenter}$, and the estimated contact area $A_{triangle}$. A second step towards making the measured force more effective is to rotate it to the direction of contact triangle normal. This is explained more in Section 5.2.2.

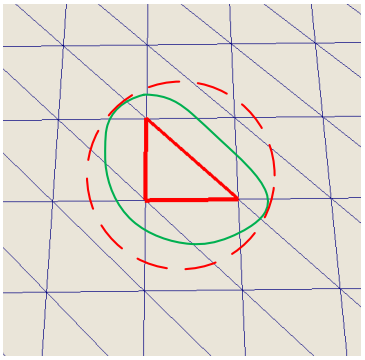


Figure 3.3: Contact triangle in red, surrounded by contact area in green. Red dashed circle is the actual contact area of the sphere indenter.

$$\boldsymbol{f_{node}} = \boldsymbol{f_{measured}} * A_{triangle}/A_{indenter} \tag{3.27}$$

The displacement of the mesh nodes are calculated using the difference between the deformed surface mesh and the initial surface mesh. The difference between each surface mesh node is then, assigned to the solid mesh nodes using barycentric coordinates. This includes all the solid mesh nodes except interior nodes whose calculations are explained in Section 3.3.2.

3.3.2 Linear Method

In our linear approach an iterative linear least square method is employed to solve for elasticity parameters. The method is then tested on data obtained from the FEM simulation in order to retrieve the parameters. In this section the linear approach is explained, and the simulation results are presented.

The model used as FEM input mesh is a tetrahedral mesh with vertices classified into four categories: interior nodes, support surface nodes, free surface nodes, and contact nodes. The distinction of these nodes are made by the force and displacement boundary conditions of them. All the nodes are free to move except support surface nodes for which zero displacement is prescribed and hence are fixed at their initial position. Amongst the free nodes, interior nodes have unknown displacement, while the displacement for the rest of the nodes are known from the surface tracking result. Considering force boundary condition, contact nodes are assigned the measured force from sensor, support surface nodes have unknown force exerted to them, and the rest have zero external force exerted to them. Table 3.1 summarizes the boundary conditions and shows the notations for the the node sets.

Nodes	Displacement		Force	
	value	notation	value	notation
Contact	tracking	$\mathbb{U}_c$	measured	$\mathbb{F}_c$
Free surface	tracking	$\mathbb{U}_f$	0	$\mathbb{F}_f$
Support surface	0	$\mathbb{U}_s$	unknown	$\mathbb{F}_s$
Interior	unknown	$\mathbb{U}_i$	0	$\mathbb{F}_i$

Table 3.1: The values and notations for each set of nodes.

In order to solve the linear least squares problem in Equation 3.26, all the nodal displacements and forces are required. However, the interior node displacements and support surface nodal forces are unknowns. Thus, the first step is to solve for unknown forces and displacements. For this purpose, the relation needs to be re-arranged in the standard form of a linear system of equations with all the unknowns as one vector. Equation 3.28 shows the relation in a matrix format for a mesh with n nodes, with each element expanded in Equation 3.29.

$$\begin{bmatrix} \boldsymbol{f_1} \\ \boldsymbol{f_2} \\ \vdots \\ \\ \\ \\ \\ \boldsymbol{f_n} \end{bmatrix}_{3n\times 1} = \begin{bmatrix} \boldsymbol{K_{11}} & \boldsymbol{K_{12}} & \dots & & \boldsymbol{K_{1n}} \\ \boldsymbol{K_{21}} & \ddots & & & \vdots \\ \vdots & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ \boldsymbol{K_{n1}} & & & & \boldsymbol{K_{nn}} \end{bmatrix}_{3n\times 3n} \begin{bmatrix} \boldsymbol{u_1} \\ \boldsymbol{u_2} \\ \vdots \\ \\ \\ \\ \\ \boldsymbol{u_n} \end{bmatrix}_{3n\times 1} \tag{3.28}$$

$$\boldsymbol{f_i} = \begin{bmatrix} f_{ix} \\ f_{iy} \\ f_{iz} \end{bmatrix}_{3\times 1}, \boldsymbol{K_{ij}} = \begin{bmatrix} k_{ij11} & k_{ij12} & k_{ij13} \\ k_{ij21} & k_{ij22} & k_{ij23} \\ k_{ij31} & k_{ij32} & k_{ij33} \end{bmatrix}_{3\times 3}, \boldsymbol{u_i} = \begin{bmatrix} u_{ix} \\ u_{iy} \\ u_{iz} \end{bmatrix}_{3\times 1} \tag{3.29}$$

As explained, all the unknown forces and displacements are arranged on the right hand side vector, while sum of the known terms of each equation is transferred to the left hand side vector. This leaves the coefficient matrix with either coefficients of unknown displacements, or zero or -1 as coefficients of unknown forces. This is illustrated in Equation 3.30 where m is the number of interior nodes with unknown displacements, p is the number of support surface nodes with unknown forces, and $\mathbf{0}$ and $\mathbf{-1}$ represent a 3×3 matrices of zeros and $-I$ respectively.

$$\begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \\ \vdots \\ \\ \boldsymbol{x_8} \\ \vdots \\ \\ \boldsymbol{x_n} \end{bmatrix}_{3n\times 1} = \begin{bmatrix} \boldsymbol{K_{13}} & \boldsymbol{K_{15}} & \dots & \boldsymbol{-1} & \boldsymbol{0} & \dots \\ \boldsymbol{K_{23}} & \boldsymbol{K_{25}} & \dots & \boldsymbol{0} & \boldsymbol{0} & \dots \\ \vdots & \vdots & & \vdots & \vdots & \\ & & & & & \\ \boldsymbol{K_{83}} & \boldsymbol{K_{85}} & \dots & \boldsymbol{0} & \boldsymbol{-1} & \dots \\ \vdots & \vdots & & \vdots & \vdots & \\ & & & & & \\ \boldsymbol{K_{n3}} & \boldsymbol{K_{n5}} & \dots & & & \end{bmatrix}_{3n\times 3(m+p)} \begin{bmatrix} \boldsymbol{u_3} \\ \boldsymbol{u_5} \\ \vdots \\ \boldsymbol{f_1} \\ \boldsymbol{f_8} \\ \vdots \\ \\ \\ \end{bmatrix}_{3(m+p)\times 1} \tag{3.30}$$

$$\boldsymbol{x_i} = \begin{cases} \boldsymbol{f_i} - \sum \boldsymbol{K_{ij}}\boldsymbol{u_j} & \boldsymbol{f_i} \notin \mathbb{F}_s \\ -\sum \boldsymbol{K_{ij}}\boldsymbol{u_j} & \boldsymbol{f_i} \in \mathbb{F}_s \end{cases}$$

$\forall j : \boldsymbol{u_j} \notin \mathbb{U}_i$

In an iterative linear method, both the unknown boundary conditions and the elasticity parameters are estimated. In this method first, initial parameter values are set to λ and μ, to build the stiffness matrix. The unknown forces and displacements are then estimated using linear least square. Having the boundary conditions for all the nodes, a new set of parameters are obtained and λ and μ are updated to the new set of parameters. The iteration continues until it converges. Algorithm 1 gives an overview of the linear method.

Foam block simulation In order to test the linear method, a discretized foam block is used as an input to the getFEM simulator. The simulator requires the traction vector applied, the contact point and boundary conditions, material parameters, and the solid mesh to provide the simulated frames. The surface mesh is generated using Meshlab software, and the solid mesh is generated by tetgen. The block dimension is $598 \times 150 \times 47.5\ mm$ in length, width, and height respectively, and consists of 265 vertices. A traction of $-50000 N/m^2$ is applied

Algorithm 1 Linear Parameter Estimation

begin

set λ, μ to initial values

re-arrange $\boldsymbol{f} = \boldsymbol{K}\boldsymbol{u}$ as Equation 3.30

for l iterations **do**

build $\boldsymbol{K}(\lambda, \mu)$

re-calculate $\boldsymbol{x_i}$ in Equation 3.30

solve linear least square (Equation 3.30) for unknown forces and displacements

solve $\left\| \begin{bmatrix} \boldsymbol{K_1 u} & \boldsymbol{K_2 u} \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \end{bmatrix} - \boldsymbol{f} \right\|_2$ for λ, μ

update λ, μ

end for

end

normal to the surface of the block, in the Y axis direction, in equal sized steps over 24 frames. Figure 3.4 shows the resulting displacements. As a result of the simulation, all the nodal forces and displacements are known, but to test the linear method, the interior node displacements and support surface forces are assumed to be unknown.

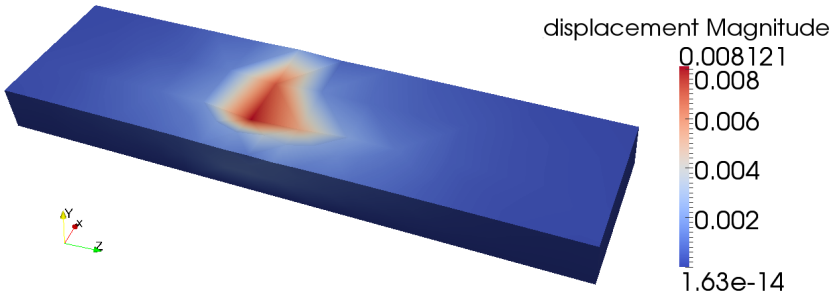


Figure 3.4: The result of linear FEM simulation on foam block. The displacement magnitude is presented in meters.

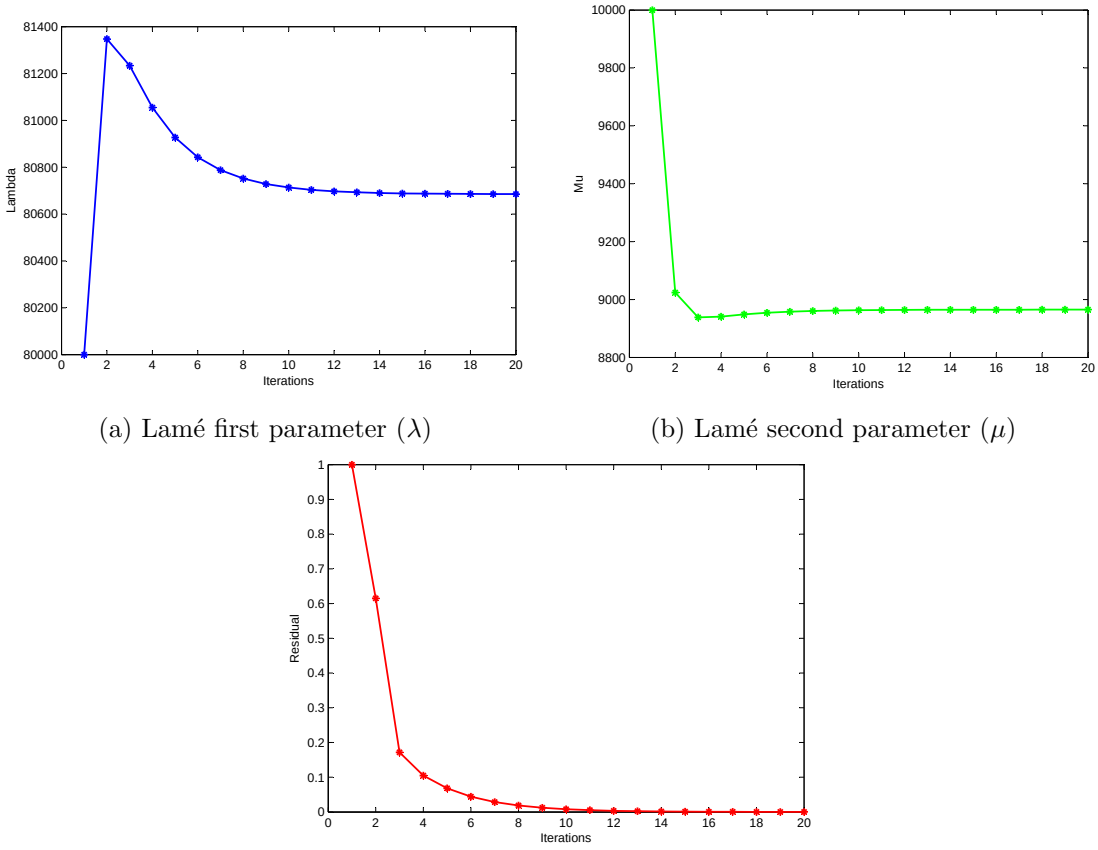
The material parameters assigned to the object in the simulation are $\lambda = 80685, \mu = 8965$. The results from the simulation are used in the linear method to retrieve the initial Lamé parameters. Figure 3.5 illustrates the convergence of λ and μ to 80685 and 8965 respectively, with the residual reduced to 10^{-4}, with the exact nodal displacements and forces employed for all the nodes except for the nodes in the interior and on the support surface.

It is important to note that the linear optimization does not seem to depend on the initial value of the parameters. As shown in Figure 3.6 five different initial values are tested to examine the convergence of the method. The method converges to the correct values every time.

The main issue with this approach is its tolerance to noise in the data and under various discretizations. Since our goal is to estimate the elasticity parameters using the displacement computed from surface tracking results, the method should be able to retrieve the parameters if the estimated approach is simulated. In order to construct the same condition, the simulated frames are fed into surface tracking and the resulting surface mesh is employed for parameter estimation. Therefore, the contact point forces are known from the simulation frames, and the surface displacement is computed using the difference between the vertex locations in the initial frame and in the deformed frame which is the result of tracking. As before, the interior node displacement and support surface forces along with elasticity parameters are unknowns. However, as Figure 3.7 illustrates, not only the estimated parameters are far from simulation parameters, but also λ has converged to a negative value which is not physical ($\lambda = -21504, \quad \mu = 21757$). Testing with tracking result shows that the method does not tolerate any deviation from the correct data, since the solid mesh discretization used in the parameter estimation is the same as the one in the simulation. Even the surface mesh which is the tracking input has the same triangulation as the surface of the model used in the simulation. For this reason we switch to another approach called displacement error minimization, which is explained in the next section.

3.3.3 Objective function

In order to improve the optimization, we need to minimize the displacement as well as the force. For this, we need to minimize the over all error $\bar{E}_U$ shown in Equation 3.31. The overall error is the weighted sum of the displacement and the force errors where u_{max} is the length of the largest nodal displacement, and f_{max} is the magnitude of the the measured force. Since Equation 3.31 is not linear, we need a non-linear optimizer like Nelder-Mead algorithm to

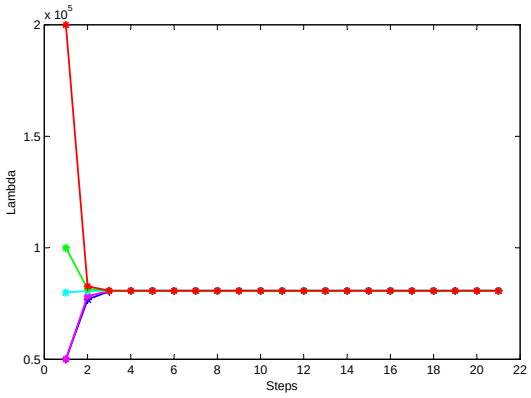


(a) Lamé first parameter (λ)

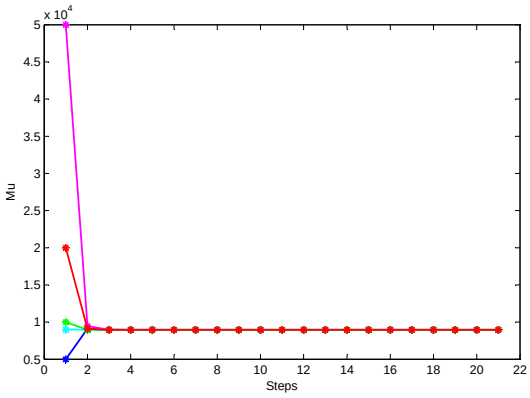
(b) Lamé second parameter (μ)

(c) Residual

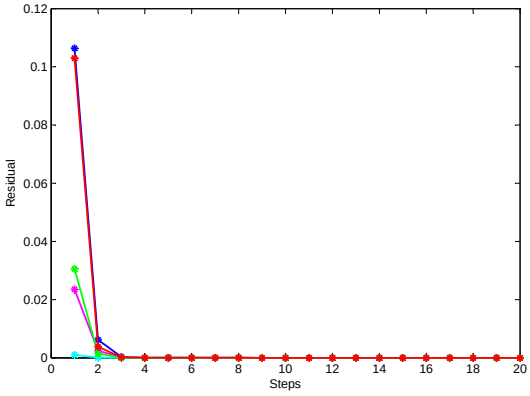
Figure 3.5: Parameter estimation result using linear method on simulation data.



(a) λ

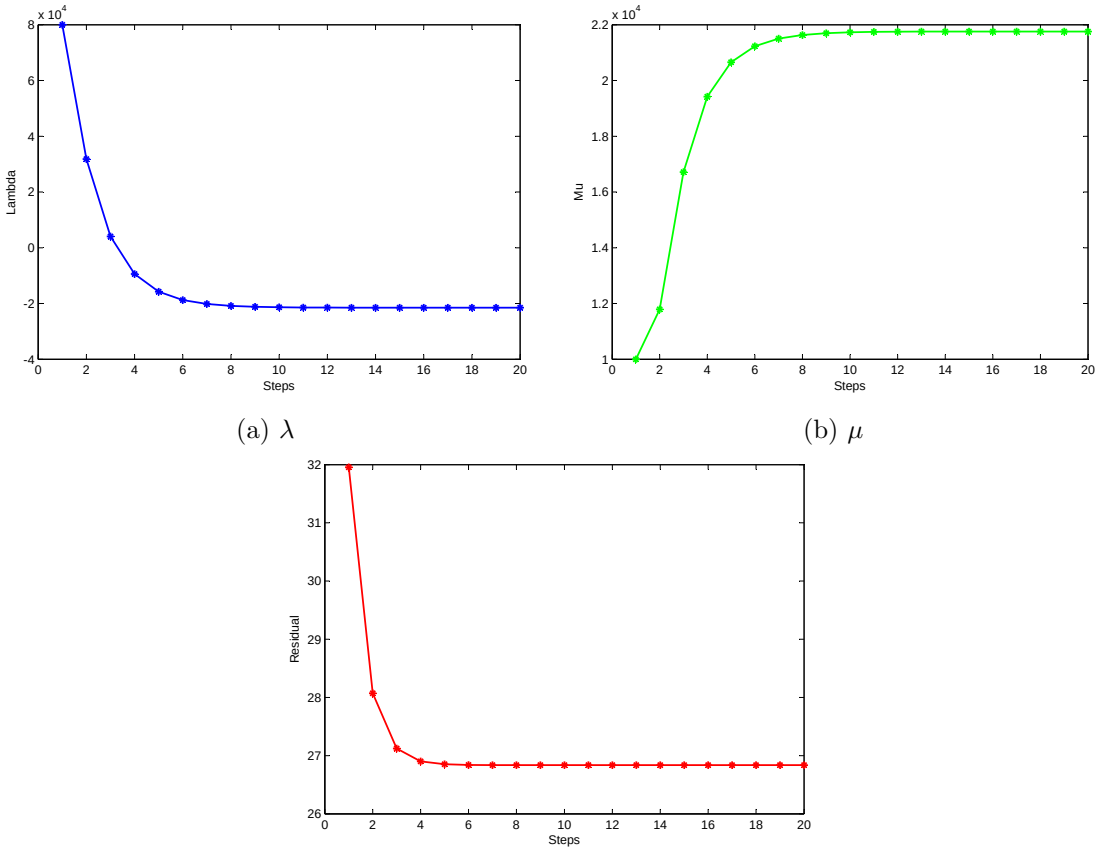


(b) μ



(c) Residual

Figure 3.6: Parameter estimation result using linear method on simulation data starting with different initial values.



(a) λ

(b) μ

(c) Residual

Figure 3.7: Parameter estimation result using linear method on simulation data and tracking result.

minimize this objective function (cost function).

$$\bar{E} = \frac{u_{max}}{f_{max}} \left\| \boldsymbol{K}\left(\lambda,\mu\right)\boldsymbol{u} - \boldsymbol{f} \right\| + \left\| \tilde{\boldsymbol{K}}^{\boldsymbol{-1}}\left(\lambda,\mu\right)\tilde{\boldsymbol{f}} - \tilde{\boldsymbol{u}} \right\| \tag{3.31}$$

The first term is the force error $\bar{E}_F = \left\| \boldsymbol{K}\left(\lambda,\mu\right)\boldsymbol{u} - \boldsymbol{f} \right\|$ and the second term is the displacement error $\bar{E}_U = \left\| \tilde{\boldsymbol{K}}^{\boldsymbol{-1}}\left(\lambda,\mu\right)\tilde{\boldsymbol{f}} - \tilde{\boldsymbol{u}} \right\|$.

In the force error term $\boldsymbol{f}$ is the measured force, and $\boldsymbol{K}\left(\lambda,\mu\right)\boldsymbol{u}$ is the calculated force $\hat{\boldsymbol{f}}$. To calculate $\hat{\boldsymbol{f}}$, Equation 3.30 is used to find the interior node displacements while other nodal displacements are given by the tracking result. Having the nodal displacements and stiffness matrix built from the given set of parameters, calculated force is obtained. Average force error is then, calculated using Equation 3.32 considering all nodes except support surface nodes, since it is not measured. According to the literature Boonvisut et al. [8] also considered both force error at the same time as displacement error in their cost function.

$$\bar{E}_F = \frac{1}{n}\sum_{i}^{n}\sqrt{(\hat{f}_{ix} - f_{ix})^2 + (\hat{f}_{iy} - f_{iy})^2 + (\hat{f}_{iz} - f_{iz})^2} \tag{3.32}$$

$$\forall i : \boldsymbol{f_i} \notin \mathbb{F}_s$$

In the displacement error term, since inverse of stiffness matrix is not computable, $\tilde{\boldsymbol{K}}^{\boldsymbol{-1}}$ is a re-arrangement similar to Equation 3.30 containing the coefficients of all the nodes except $\mathbb{U}_s$ which is equal to zero and the coefficient of -1 for $\mathbb{F}_s$. This means it considers all the nodal displacements as unknowns except support surface nodes. Therefore $\tilde{\boldsymbol{u}}$ is a mixture of displacements $(\mathbb{U}_c, \mathbb{U}_f, \mathbb{U}_i)$ and forces $(\mathbb{F}_s)$, and $\tilde{\boldsymbol{f}}$ is $\boldsymbol{x_i}$. Thus the displacement part of $\tilde{\boldsymbol{u}}$ is called $\hat{\boldsymbol{u}}$ and the average of the displacement error is calculated considering only surface nodes since there are no measurements for interior nodes. Equation 3.33 calculates the amount of deviation from the surface tracking results u.

$$\bar{E}_U = \frac{1}{n}\sum_{i}^{n}\sqrt{(\hat{u}_{ix} - u_{ix})^2 + (\hat{u}_{iy} - u_{iy})^2 + (\hat{u}_{iz} - u_{iz})^2} \tag{3.33}$$

$$\forall i : \boldsymbol{u_i} \notin \mathbb{U}_i$$

3.3.4 Displacement Error Minimization Method

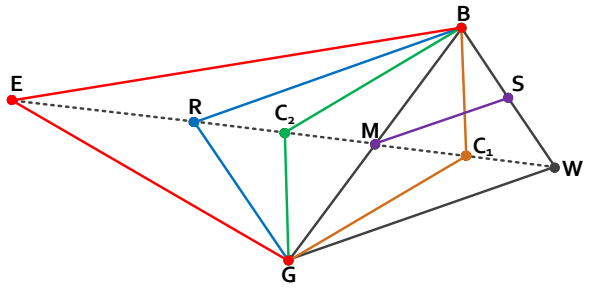


Figure 3.8: Nelder-Mead simplex algorithm geometry.

The displacement error minimization approach uses the Nelder-Mead simplex algorithm in order to optimize the elasticity parameters. This approach is employed since the derivative of problem cannot be determined. This algorithm finds a local minimum of a cost function with several variables. In this case there are two variables (λ and μ), the simplex in 2D is a triangle, and the method performs a comparison between the values of the cost function at the three vertices of the triangle. The vertex with largest cost function value is considered as worst vertex, thus it is replaced by a new vertex to form a new triangle. In this way, the method continues to find a series of triangles, for which the value of the cost function decreases continuously. The size of the triangle also, decreases and the coordinates of the triangle are found as the solution to the minimization problem. Figure 3.8 shows the choices for replacing the worst vertex ($\boldsymbol{w}$) while keeping the good ($\boldsymbol{g}$) and best ($\boldsymbol{b}$) vertices. The procedure is explained in Algorithm 2.

In the displacement error minimization approach, it is important to select the initial values appropriately, so that it is not trapped in a local minima. Having the contact point displacement and force measurements, one can compute the reduced Young's Modulus from the Hertzian model. By considering a grid of Poisson's ratios within the range of 0.05 to 0.45 with steps of 0.05, the corresponding Young's Modulus is obtained. The equivalent Lamé parameters are then computed. Each of these Lamé parameters are used as inputs to the cost function, and finally the one with the lowest cost function value is selected as initial value to the optimization.

Algorithm 2 Nelder-Mead simplex algorithm

begin

get 3 points: $\boldsymbol{b} = (\lambda_1, \mu_1), \boldsymbol{g} = (\lambda_2, \mu_2), \boldsymbol{w} = (\lambda_3, \mu_3)$

while $f(\boldsymbol{b})$ larger than threshold **do**

sort by function values and re-assign such that $f(\boldsymbol{b}) \geq f(\boldsymbol{g}) \geq f(\boldsymbol{w})$

$\boldsymbol{m} = \frac{1}{2}(\boldsymbol{b} + \boldsymbol{g})$, $\boldsymbol{r} = \boldsymbol{m} + (\boldsymbol{m} - \boldsymbol{w})$, $\boldsymbol{e} = \boldsymbol{r} + (\boldsymbol{r} - \boldsymbol{m})$

if $f(\boldsymbol{e}) < f(\boldsymbol{w})$ **then**

$\boldsymbol{w} = \boldsymbol{e}$ continue the loop

end if

if $f(\boldsymbol{r}) < f(\boldsymbol{w})$ **then**

$\boldsymbol{w} = \boldsymbol{r}$ continue the loop

end if

$\boldsymbol{c_1} = \frac{1}{2}(\boldsymbol{m} + \boldsymbol{w})$, $\boldsymbol{c_2} = \frac{1}{2}(\boldsymbol{m} + \boldsymbol{r})$

if $f(\boldsymbol{c_1}) < f(\boldsymbol{w})$ **then**

$\boldsymbol{w} = \boldsymbol{c_1}$ continue the loop

end if

if $f(\boldsymbol{c_2}) < f(\boldsymbol{w})$ **then**

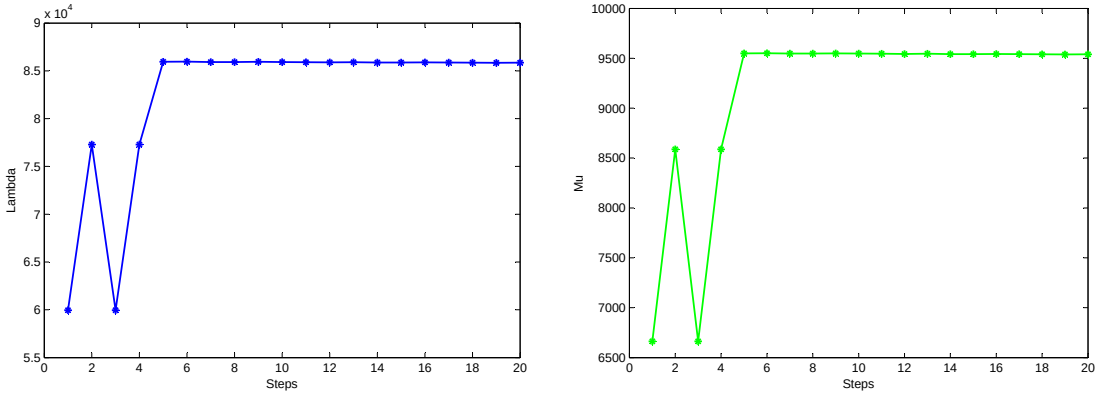
$\boldsymbol{w} = \boldsymbol{c_2}$ continue the loop

end if

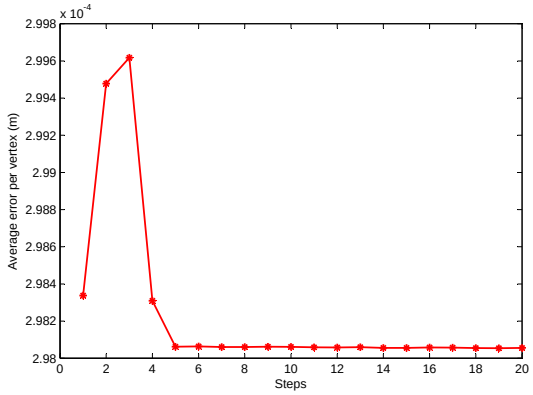
$\boldsymbol{w} = \frac{1}{2}(\boldsymbol{b} + \boldsymbol{w})$, $\boldsymbol{g} = \frac{1}{2}(\boldsymbol{b} + \boldsymbol{g})$

end while

end



(a) λ (b) μ



(c) Error

Figure 3.9: Parameter estimation result using displacement error minimization method on simulation data.

Foam block simulation In order to test the displacement error minimization method, the same foam block simulation results are used. The contact forces are known from simulation result, and the surface displacement are computed using the tracking result. Figure 3.9 shows the optimization using distance error only and the parameters converge to $\lambda = 85252$ and $\mu = 9472$. Although they did not converge to the right value exactly, the results have improved compared to previous method. Figure 3.10 demonstrates the average displacement error per node. It shows some mismatch on one of the contact nodes, which explains the deviation from the exact values.

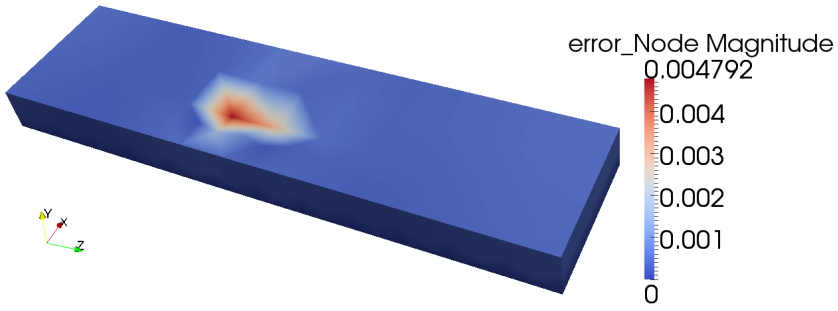


Figure 3.10: The mismatch between measured deformation and computed deformation using linear FEM. The magnitude of nodal error is presented in meters.

3.4 Summary

In this chapter the theoretical concepts of continuum mechanics and the Finite Element Method were presented. The two approaches used for elasticity parameter estimation are then explained and tested for a foam block simulation data. The linear method shows exact convergence for the correct data, but when it comes to input data with some error, it fails to estimate the right parameters. Therefore, the displacement error minimization method is introduced. The method minimizes a cost function which tries to bring the calculated data close to the measured data both for displacement and force. The test result for this method shows improvement compared to the former method, though it did not estimate the exact values.

Chapter 4

Physical Properties Acquisition System

Having accurate sensor data is important for estimating the correct parameters of the object. Therefore, the data acquired is analyzed to have an acceptable amount of error. In Section 4.1, we describe the device setup, data acquisition process, and the relation between coordinate frames. For making the visual input data ready to use, some pre-processing steps need to be taken, including background removal, contact point tracking, and workspace error controlling, which are discussed in Section 4.2. Section 4.3 explains the surface tracking method employed to track the soft object.

4.1 Acquisition System

In this section we present the acquisition system components employed for the elastic parameter estimation framework. The purpose and function of each device is described along with the synchronization of data. First the depth sensors are presented, afterwards the force sensor is introduced, and lastly registering all the reference frames of the sensors into a common frame is discussed.

Figure 4.1: Depth Sensors.

4.1.1 Range Scanner

The 3D laser scanner (VIVID 910) is used to generate accurate models of colored 3D objects (Figure 4.1). It employs laser beam light-sectioning technology which scans the object using a slit beam. Light reflected from the object is acquired by a charge-coupled device (CCD) camera and 3D data is then created by triangulation to determine distance information. It acquires the color of the object, with the same CCD as used for distance data, by employing a rotating filter to separate the acquired light. The accuracy of the VIVID 910 given by the manufacturer as $\pm 0.22\ mm$ on the x-axis, $\pm 0.16\ mm$ on the y-axis and $\pm 0.1\ mm$ on the z-axis. The scanner obtains up to 640×480 individual points per scan. The tele and middle lens of the scanner are used for small and medium size objects respectively, with measurement distance from 600 mm to 2500 mm with auto focus. The object to be modeled was placed within a distance of around 700 mm from the scanner in the setup of this thesis.

This sensor is used to acquire a 3D model of an object by recording it from different view points. The view points are then registered and merged using Geomagic Studio software to generate a watertight model, which is used later as the undeformed template mesh for surface

tracking and parameter estimation.

4.1.2 Depth Sensors

The deformation sequence is recorded by series of color and depth images acquired. One of two depth sensors are employed to generate the object models: a Point Grey Research Bumblebee2 stereo-head and a popular Kinect sensor for Xbox 360. The following section elaborates the function and model reconstruction by each sensor.

Kinect Sensor

Kinect for Xbox 360 sensor is a Microsoft product introduced for the purpose of real time gesture detection based on medium quality textured 3D data. The sensor consists of a color camera, an infrared camera, an infrared projector, a motorized tilt mechanism, and a multi-array microphone to provide interaction via voice commands (Figure 4.1). It is capable of streaming data at a frame rate of 30 frames per second with a resolution of 640×480. The output image of the color camera is three 8-bit channels, while for the IR camera the output image is a 10-bit single channel image. The color camera covers a larger field of view compared to IR camera. The purpose of IR camera is to detect the IR projected pattern in order to estimate the depth. A 830 nm laser diode with a wavelength that is not visible to the human eye, in the IR projector illuminates the scene, therefore the projected pattern created is in the infrared spectrum and remains invisible too. An optical filter is built into the IR camera which is sensitive to laser diode wavelength, to generate a crisp pattern of the projection over the IR image plane. The pixels in the depth image represent the depth of the closest surface in the scene, mapped to that pixel with respect to camera optical center. The maximum resolution of the depth image supported by the Kinect is 640×480 pixel and its field of view is the same as IR camera. A 11-bit disparity value is found for each depth image pixel [46].

The Kinect sensor operates on the principle of a structured light range sensor. A fix and predefined pattern is generated by projecting light from a laser diode through a transparency micro-grid containing the engraved pattern. The IR camera captures the reflected light pattern

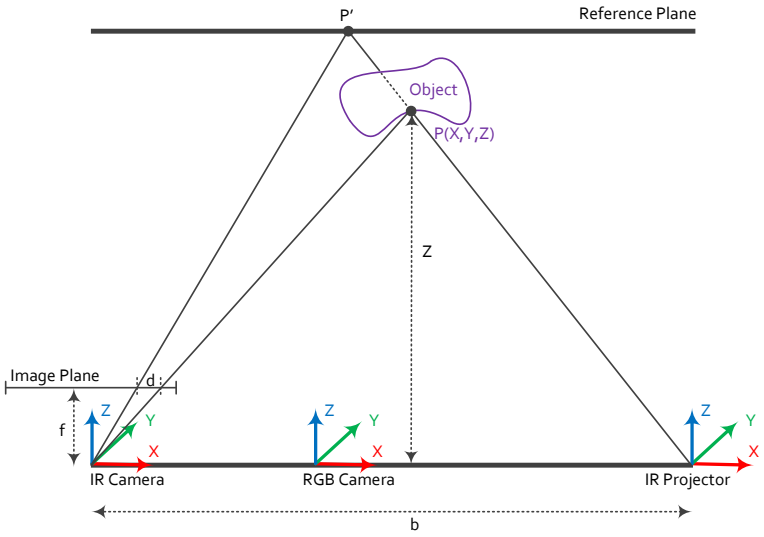


Figure 4.2: Kinect Camera Model.

on the surface of the objects and compares it against the predefined reference pattern [25]. Figure 4.2 illustrates the triangulation process used for depth calculation. Point P on an object is in depth Z from the optical center of the infrared camera. The axis of the IR camera is orthogonal to the image plane and passes through the optical center in the direction of the object. The IR projector is aligned with the x-axis of the IR camera. The distance between the optical center of the IR camera and that of the IR projector is defined by the baseline (b). The reference plane in the diagram of Figure 4.2 corresponds to the projected IR image as a reference. This reference image contains a predefined pattern which consists of tiny dots imaged in the IR spectrum and its reference distribution is stored in the firmware of the sensor during the manufacturing process. When an object is placed in the field of view of the Kinect depth sensor, the reference pattern captured by the IR camera is deformed. The resulting shift of the dots in the pattern from the reference image is measured as a disparity in the image plane of the IR camera. The shift solely takes place along the x-axis since the IR camera and the IR projector are only translated along the x-axis, by the baseline distance [46]. Equation 4.1 defines the relationship between the distance to the object, Z, and the raw disparity, d.

$$Z = \frac{fb}{d} \tag{4.1}$$

where Z is the depth of the object along the z-axis of the IR camera from its optical center (in meters), b is the horizontal baseline between the IR camera and the IR projector (in meters), f is the focal length of the IR camera (in pixels), and d is the raw disparity (in pixels). The raw disparity is calculated by finding the correlation between every point on the reference pattern, P', and the corresponding point on the deformed pattern, P, using a window of size 9×9 pixels [67]. The raw disparity is then used to provide the normalized disparity, d', between 0 and 2047 (11-bit integer). Equation 4.2 defines the relationship between the raw and normalized disparity, (d and d'), which is replaced in Equation 4.1 to estimate the depth of a given point (Equation 4.3).

$$d = md' + n \tag{4.2}$$

$$Z = \frac{fb}{md' + n} \tag{4.3}$$

Where m and n are factors for denormalization [35]. In principle, Kinect provides 2048 levels of disparity due to its 11-bit value. However, the Kinect for Xbox 360 returns disparity values between 400 and 1050, which restrict the output between 0.5 m and 8 m. [46]. In this thesis the OpenNI framework is used with the Kinect to provide data between 0.5 m and 8 m.

Bumblebee2 Stereo Camera

The color Point Grey Research Bumblebee2 color stereo-head is a commercial stereo vision camera with two high quality and calibrated camera lenses. This commercial stereo camera is employed during the deformation procedure to generate high quality and dense point cloud.

The stereo camera operates on the principle of triangulation on images captured from different points of view of the same scene [55]. As shown in Figure 4.3, left and right cameras capture

two images from optical centers O_L and O_R, respectively. Points P_L and P_R are left and right image points correspondent to object point P. The triangulation finds the object point P by intersecting the rays that pass from left and right origin to the object point P. The position of point P can be calculated based on image points P_L and P_R and the camera intrinsic and extrinsic parameters.

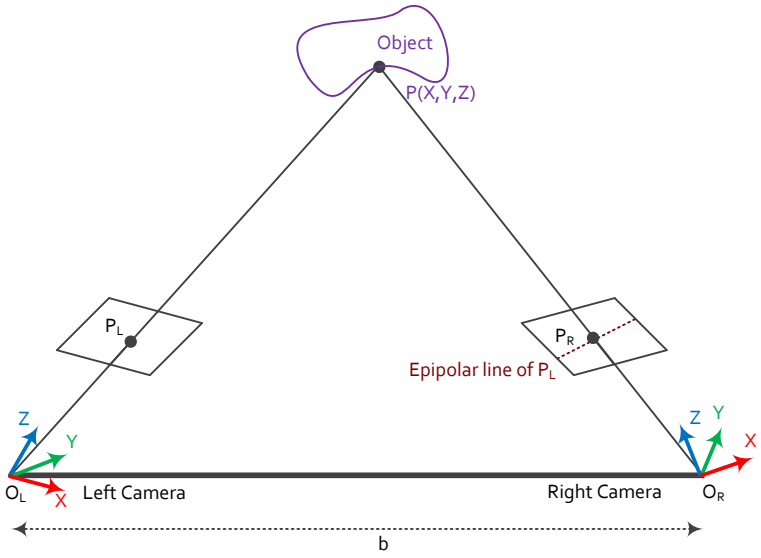


Figure 4.3: Stereo Camera Model.

In order to be able to calculate the 3D position of point P, its corresponding image points have to be determined first. Having the image point P_L on the left image plane, the position of image point P_R is restricted on a line known as epipolar line. This line can be calculated by projecting the ray from left camera origin to the object point P, into the right image plane. Therefore, the search for P_R is restricted to the epipolar line of P_L.

The deformation sequences were captured with the Bumblebee2 camera at a resolution of 1024×768 at 24 frames per second. The Point Grey Research Triclops library was used to calculate the point cloud with the available window-based correlation method with a window size of 11×11 and points filtered with "back-and-forth" [26] and the "surface validation" [54] method. The back and forth validation verifies that the matched points are same when the

reference image is the right image or the left image. The surface validation method removes spikes caused by feature mismatches by segmenting the image into disparity surfaces and deciding on a point according to the surface size it belongs to. The Bumblebee2 camera only provides 30 disparity levels between 0.5 m and 1 m of range, due to its 8 bit disparity value. However, using the Triclops library one can enable sub pixel interpolation and use the 16 bit value. The Bumblebee2 camera has the advantage of a higher resolution and gives good results with textured objects.

Vertex Normal Estimation

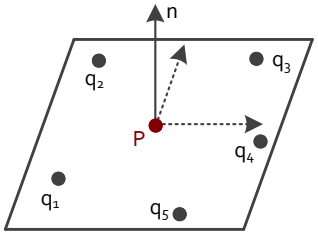


Figure 4.4: A plane is fitted to vertex P and its neighbors.

A local plane fitting method is performed on the point cloud obtained from the one of the sensors described earlier to calculate the normal of vertices [29]. The vertex normals are orthogonal vectors to a tangential plane containing that vertex and its neighbors (Figure 4.4). Therefore we need to find a plane that is as close as possible to a set of n points $(\boldsymbol{p_1}, \boldsymbol{p_2}, ..., \boldsymbol{p_n})$. The distance from points to plane is defined by the square sum of orthogonal distance between points and the plane. If we assume that c is a point on the plane, and n is the normal to that plane, the orthogonal distance between a point $\boldsymbol{p_i}$ and the plane is $(\boldsymbol{p_i} - \boldsymbol{c})^T \boldsymbol{n}$. Thus we need to minimize Equation 4.4

$$\underset{c,\|n\|=1}{min} \sum ((\boldsymbol{p_i} - \boldsymbol{c})^T \boldsymbol{n})^2 \tag{4.4}$$

By solving the above equation, point $\boldsymbol{c}$ on the plane is found as $\boldsymbol{c} = \frac{1}{n} \sum \boldsymbol{p_i}$. Equation 4.4 can be reformulated as

$$\min_{\|n\|=1} \left\|\boldsymbol{A^T n}\right\|_2^2 \tag{4.5}$$

where $\boldsymbol{A} = [\boldsymbol{p_1 - c, p_2 - c, ..., p_n - c}]$ is a $3 \times n$ matrix. Using the singular value decomposition ($\boldsymbol{A} = \boldsymbol{USV}^T$) of A in the optimization, it can be written as

$$\left\|\boldsymbol{A^T n}\right\|_2^2 = \left\|\boldsymbol{V S^T U^T n}\right\|_2^2 = \left\|\boldsymbol{S^T U^T n}\right\|_2^2 = (\sigma_1 \boldsymbol{y_1})^2 + (\sigma_2 \boldsymbol{y_2})^2 + (\sigma_3 \boldsymbol{y_3})^2 \tag{4.6}$$

where $\boldsymbol{y} = \boldsymbol{U^T n}$ is the unit vector and σ_i are singular values of $\boldsymbol{A}$. The minimal value for above equation is σ_3^2 which is the smallest singular value in $\boldsymbol{S}$. Therefore Equation 4.6 is minimized for $\boldsymbol{y} = (0, 0, 1)^T$ or equivalently the third column of $\boldsymbol{U}$. In this way the normal vector of the vertex is obtained directly.

4.1.3 Force Sensor

An ATI Industrial Automation force/torque sensor Nano25 is used to measure the force exerted on the object. The sensor system consists of a transducer, shielded cable, and an Ethernet/device net interface (Figure 4.5). This sensor is capable of measuring forces up to 125 N on the X and Y axis, and 500 N on the Z axis and torques up to 3 Nm with all three degrees of freedom. The force sensor outputs are recorded via a National Instrument PCI-6023E multifunction data acquisition system. The sensor is mounted on a machined sphere head indenter with a handle, which can be used free-hand.

In order to obtain meaningful data, the device needs to be calibrated using the calibration file provided in the commercial package. ATICombinedDAQFT.NET class library is used to record either single data points or continuous data. However, before starting to record the data, one should bias out the current load applied to the transducer. The transducer acquires the voltage samples at a frequency rate of $1000Hz$, it then applies an averaging filter of size 10, to yield an actual sampling rate of $100Hz$. The measured voltages are converted to force and torque measurements with the help of the calibration values.

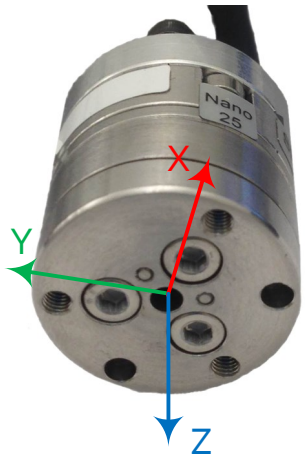
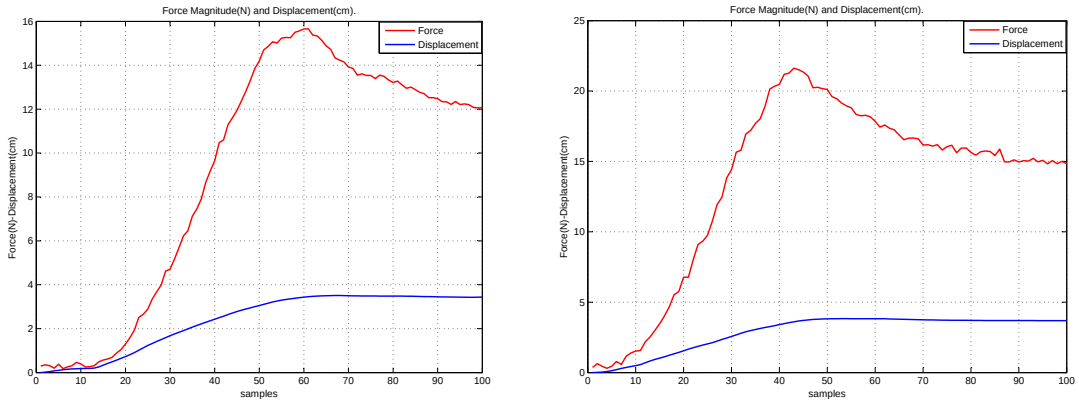


Figure 4.5: ATI Industrial Automation force/torque sensor Nano25.

During the deformation procedure, a program records the force readings along with its time stamp at the same time as one of the depth sensors are recording the scene. When the procedure terminates, a comparison between the image and force reading time stamps is performed, and the closest reading with respect to time, is chosen as the corresponding force measurements for that scene. Figure 4.6 illustrates the synchronization between visual tracking and force recording via a foam loading test. The force sensor is employed to indent a foam block gradually and keep the position constant at a certain depth. As shown in the Figure 4.6, the maximum force is exerted on foam when maximum displacement is achieved, then for constant displacement the force magnitude starts to drop due to viscoelastic effect of the object.

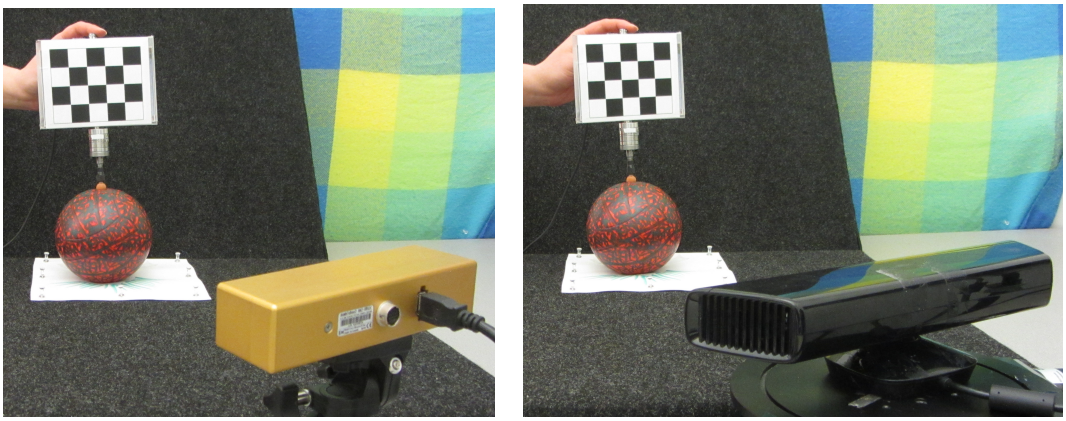
4.1.4 Calibration

In order to perform visual tracking and transform measured values to a single coordinate frame, the transformation between the coordinate frames of the devices needs to be computed. Figure 4.7 demonstrates the acquisition setup for the Bumblebee2 and Kinect respectively. The first step is to calibrate the cameras so that the relation between a 3D scene point and a 2D image point is estimated. This relation is formed by intrinsic and extrinsic camera parameters. The Bumblebee2 does not require this calibration procedure as the manufacturer has included the precise parameters with the product. In contrast, the calibration procedure is performed



(a) Bumblebee (b) Kinect

Figure 4.6: Force-Displacement synchronization test via loading a foam block.



(a) Bumblebee (b) Kinect

Figure 4.7: Data capturing setup.

for the Kinect sensor to obtain the precise internal parameters of each built in camera and determine the position and orientation of one camera with respect to the other. The next paragraphs elaborate on the camera calibration for Kinect sensor, followed by the relationship of other coordinate frames in the setup.

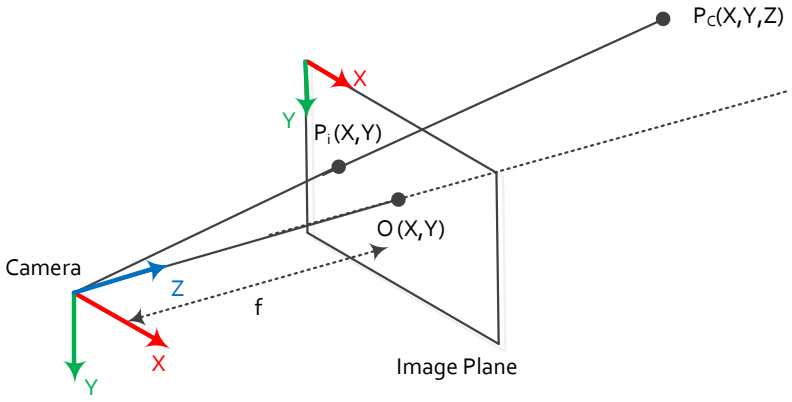


Figure 4.8: Pinhole camera model.

As mentioned, the goal of camera calibration is to estimate the relationship between a scene point and an image point. The pinhole camera model defines this relationship based on focal length of camera (the distance between the center of projection and the image plane) and its principal point (the intersection of the image plane and the optical axis). As shown in Figure 4.8, the object's position is defined with respect to camera reference frame in world units, while the image position is defined with respect to image reference frame in pixel unit. P_c is the real world object point with respect to camera reference frame, and P_i is the location of that point on the image plane with respect to image reference frame. Tsai [72] defines the pinhole camera model as in Equations 4.7, which relates real world points, P_c, in world unit to their 2D projections, P_i, on the image plane in pixel units. In this relation f is the focal length of the lens, (s_x, s_y) are the size of one pixel, and (O_x, O_y) are the optical center on the image plane.

$$\boldsymbol{K} = \begin{bmatrix} \frac{f}{s_x} & 0 & O_x \\ 0 & \frac{f}{s_y} & O_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.7}$$

In reality, lens distortions (radial distortion (k_1, k_2, k_3)) and/or optical axis misalignment (tangential distortion (p_1, p_2)) cause the simple pinhole camera model to fail to find a perfect correspondence between the 2D image plane and the 3D world coordinates. These distortions parameters convert the perspective projection relationship into a nonlinear form.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \boldsymbol{K} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}, \quad \boldsymbol{p_i} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \frac{x_1}{x_3} \\ \frac{x_2}{x_3} \end{bmatrix} \tag{4.8}$$

The other set of parameters known as the extrinsic parameters, define the relationship between the world reference frame and the camera coordinate system. Since the optical center of the camera or the location of the camera frame is placed at an unknown location inside the camera, it is difficult to define the location of an object point with respect to the camera frame. Therefore the point is first defined in the world coordinate system and then transformed to the camera frame. The transformation between world coordinate and the camera coordinate system is defined using Equation 4.9.

$$\boldsymbol{p_c} = \boldsymbol{R}_{3\times3}.\boldsymbol{p_w} + \boldsymbol{t}_{3\times1} \tag{4.9}$$

where $\boldsymbol{R}$ is a rotation matrix of size 3×3 and $\boldsymbol{t}$ is a 3×1 translation vector. Using this equation, the object point in the world coordinate system, $\boldsymbol{p_w}$, is first transformed into the camera coordinate system, $\boldsymbol{p_c}$, then the relationship between this point and its image coordinates in pixels, is defined by Equation 4.7 and 4.8.

The calibration procedure for Kinect estimates the intrinsic and extrinsic parameters of each IR and RGB cameras, as well as the extrinsic parameters between the IR and RGB cameras.

(a) Color (b) IR

Figure 4.9: Checkerboard pattern used to calibrate the Kinect.

Since these parameters are fixed, the procedure is performed only once. The estimation of the intrinsic parameters for the color and the IR cameras, include the focal length (f_x, f_y) , the principal point (O_x, O_y), and the lens distortion coefficients $(k_1, k_2, p_1, p_2, k_3)$ [76]. A regular checkerboard target of size 10×7 (Figure 4.9) fixed at a board is employed to calibrate the IR and RGB cameras of the Kinect. During the calibration procedure the IR projector is covered to prevent the noise it produces on the IR image. Since without the projection, the IR image is too dark for the feature points to be retrieved, a standard lamp illuminates the target board. Figure 4.9 shows the IR and the corresponding color image. Fifty calibration images showing different poses of the checkerboard pattern covering the workspace are captured in a way that it is in the field of view and spectrum of both IR and RGB cameras. The 2D corresponding points of each view of the checkerboard are extracted by OpenCV `findChessboardCorners` function. This function gives the location of the corners on the image plane. All the 2D projection points along with the 3D object points are then used to calculate the intrinsic parameters of camera using OpenCV `calibrateCamera` method. This method estimates the intrinsic and extrinsic parameters for each view of the checkerboard using Zhang's camera calibration method [76]. The method also calculates and minimizes the re-projection error E_r that is the sum of the square distance between 2D image points (obtained from corners extraction) and projected 2D

points using the current estimation of intrinsic and extrinsic parameters.

$$E_r = \sum_i [(x_i - x_i^{'})^2 + (y_i - y_i^{'})^2] \tag{4.10}$$

Where (x_i, y_i) and $(x_i^{'}, y_i^{'})$ denote the image points and the projected points respectively. Table 4.1 shows the intrinsic parameters of the color and the IR camera resulting from the calibration. The parameters of IR camera are not used in visual tracking process.

Camera	F_x	F_y	C_x	C_y
Color	522.5925	520.5950	310.7083	233.2976
IR	584.6213	583.1574	323.8129	249.3762

Table 4.1: Focal length and image center of the color and IR camera of the Kinect.

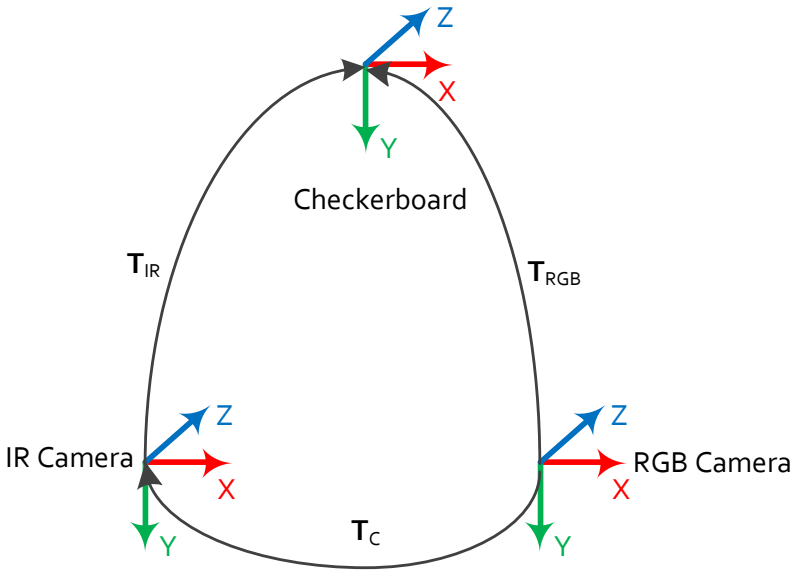


Figure 4.10: Stereo calibration in Kinect.

The camera calibration method proposed by Zhang's [24] also provides the pose estimation of the checkerboard target with respect to a camera frame as shown in Figure 4.10. For the fixed target in both camera views, the geometrical transformation between the cameras is defined by

Equation 4.11.

$$\boldsymbol{T_{IR}} = \boldsymbol{T_C^{-1}}\ \boldsymbol{T_{RGB}} \tag{4.11}$$

where $\boldsymbol{T_C}$ is the homogenous transformation matrix (consists of rotation matrix, and translation vector) from the IR camera to the RGB camera, $\boldsymbol{T_{IR}}$ is the homogenous transformation matrix from the IR camera to the checkerboard target, and $\boldsymbol{T_{RGB}}$ is the homogenous transformation from the RGB camera to the checkerboard target. This can be determined by stereo calibration. Equation 4.12 shows transformation matrix $\boldsymbol{T_C}$ which is the result of stereo calibration between the IR and RGB sensors. The camera calibration result depends on experimental conditions such as the quality of calibration images, flatness of the checkerboard pattern picture, and the number of calibration images.

$$\boldsymbol{T_C} = \begin{bmatrix} 0.99998109 & -0.00540787 & 0.00292595 & -2.50043218 \\ 0.00537898 & 0.99993757 & 0.00979367 & 0.00701323 \\ -0.00297873 & -0.00977775 & 0.99994775 & 0.27232608 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.12}$$

4.2 Pre-processing

4.2.1 Background removal

The goal of background subtraction is to subtract the current image from a reference background image. Since the background in our scene is static, a simple background model can be used that models the intensity value of a pixel. In this thesis, we use a statistical approach proposed by Horprasert et al. [30] which is capable of dealing with local illumination change such as shadows as well as global illumination change. Their method is based on separation of brightness and chromaticity of a pixel. Figure 4.11 illustrates their color model in RGB space. The method consists of two steps: building the background model using 4 values; and measuring the deviation of current image pixels from background model and classifying the pixels.

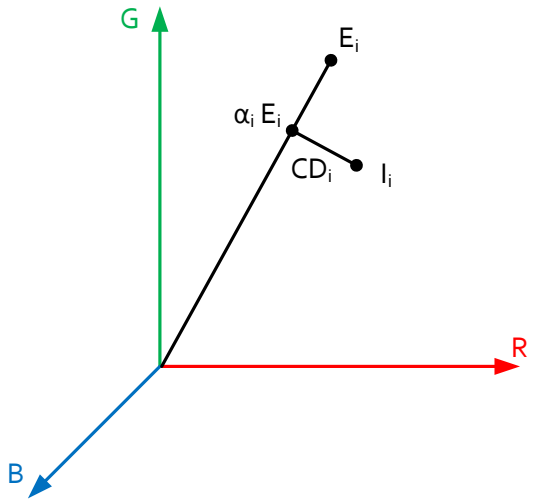


Figure 4.11: Color model used for background subtraction.

Figure 4.11 illustrates how the background model is built for each pixel over N background frames. $\boldsymbol{E_i} = [E_R(i); E_G(i); E_B(i)]$ is the expected RGB color for pixel i over N frames, which makes the line $\boldsymbol{OE_i}$ the chromaticity line of that pixel. $\boldsymbol{I_i} = [I_R(i); I_G(i); I_B(i)]$ is the current background image color values for pixel i. The goal is to measure the distortion of pixel i in current image from $\boldsymbol{E_i}$, but the distortion is decomposed into brightness and chromaticity distortion measurements. The brightness distortion α is a scalar value defined such that the distance between the current image pixel value and its projection on the expected chromaticity line is minimized. Since the projected value has the same chromaticity but different brightness. α_i is equal to one if the current image pixel has the same brightness of background model, less than one if it is darker, and more than one if it is brighter. The chromaticity distortion is the orthogonal distance between the current image pixel and the expected chromaticity line in the color space.

As mentioned earlier, each pixel of the background model is modeled by 4 values: $< \boldsymbol{E_i}, \boldsymbol{S_i}, a_i, b_i >$ where $\boldsymbol{E_i}$ is the expected color value, $\boldsymbol{S_i}$ is the standard deviation of color value, a_i is the variation of the brightness distortion, and b_i is the variation of the chromaticity distortion of the i th pixel. E_i (Equation 4.13) and $\boldsymbol{S_i}$ (Equation 4.14) are respectively mean and standard deviation

of i th pixel in each color channel over N background frames. Although the background images are static, the pixel values are different for N frames due to camera noise and illumination fluctuations by the light source. According to their definition, brightness and chromaticity distortion are determined by Equation 4.15 and 4.16, respectively. This division by standard deviation is to normalize the distortion calculation.

$$\boldsymbol{E_i} = [\mu_R(i), \mu_G(i), \mu_B(i)] \tag{4.13}$$

$$\boldsymbol{S_i} = [\sigma_R(i), \sigma_G(i), \sigma_B(i)] \tag{4.14}$$

$$\phi(\alpha_i) = (\boldsymbol{I_i} - \alpha_i \boldsymbol{E_i})^2 \tag{4.15}$$

$$\alpha_i = min\left[\sum_{C=R,G,B}\left(\frac{I_C(i)-\alpha_i\mu_C(i)}{\sigma_C(i)}\right)^2\right] = \frac{\left(\frac{I_R(i)\mu_R(i)}{\sigma_R^2(i)}+\frac{I_G(i)\mu_G(i)}{\sigma_G^2(i)}+\frac{I_B(i)\mu_B(i)}{\sigma_B^2(i)}\right)}{\left(\left[\frac{\mu_R(i)}{\sigma_R(i)}\right]^2+\left[\frac{\mu_G(i)}{\sigma_G(i)}\right]^2+\left[\frac{\mu_B(i)}{\sigma_B(i)}\right]^2\right)}$$

$$CD_i = \|\boldsymbol{I_i} - \alpha_i \boldsymbol{E_i}\| = \sqrt{\sum_{C=R,G,B}\left(\frac{I_c(i)-\alpha_i\mu_c(i)}{\sigma_c(i)}\right)^2} \tag{4.16}$$

After calculating the distortion measurements for each background frame, the variation of brightness and chromaticity distortion needs to be calculated over space and time. a_i and b_i represent the variation of brightness and chromaticity distortion respectively.

$$a_i = RMS(\alpha_i) = \sqrt{\frac{\sum_{i=0}^{N}(\alpha_i - 1)^2}{N}} \tag{4.17}$$

$$b_i = RMS(CD_i) = \sqrt{\frac{\sum_{i=0}^{N}(CD_i)^2}{N}}$$

The next step is to classify the image containing the foreground image, by evaluating the difference between the background model and the foreground image. For this purpose, the brightness and chromaticity difference is calculated using Equations 4.15 and 4.16 and normal-

ized by dividing by distortion variations using Equation 4.18 so that a single threshold could be used for all pixels.

$$\hat{\alpha_i} = \frac{\alpha_i - 1}{a_i} \tag{4.18}$$

$$\hat{CD_i} = \frac{CD_i}{b_i}$$

Finally, a pixel of foreground image is classified as foreground if it has chromaticity difference from expected value of the background model, otherwise it is considered background. The threshold is selected empirically for the scene.

$$\begin{cases} Foreground: & \hat{CD_i} > \tau_{CD} \\ Background: & otherwise \end{cases} \tag{4.19}$$

4.2.2 Marker Tracking

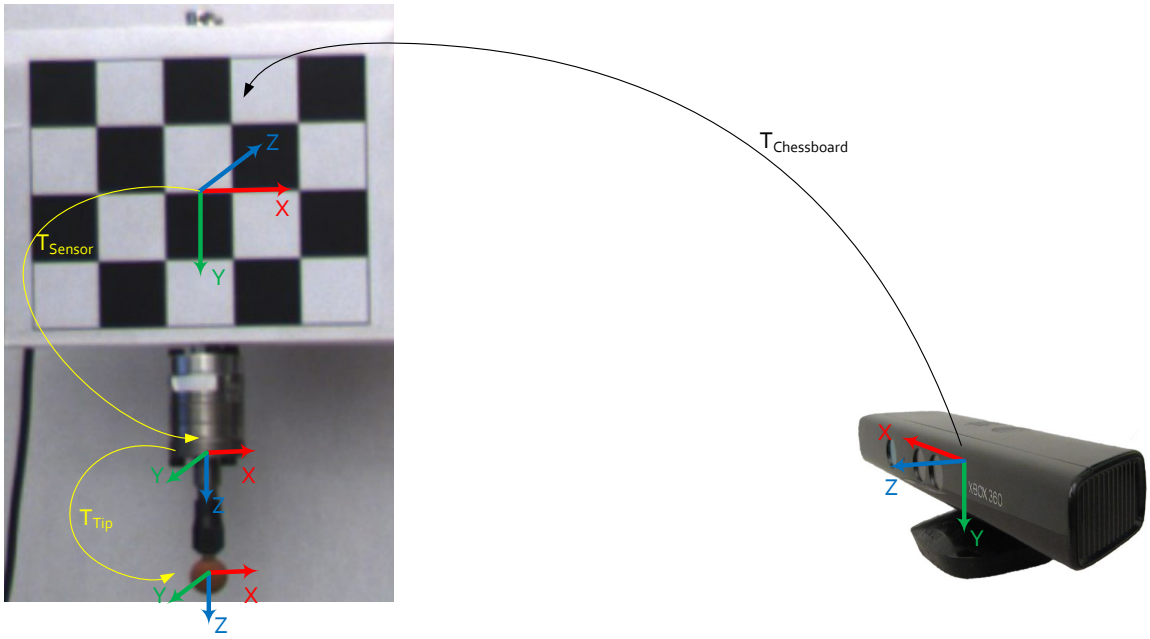


Figure 4.12: Transformation from force sensor tip to camera frame.

For estimating the position and orientation of the force sensor probe, a 4×5 checkerboard is

fixed on the probe handle such that its X axis is aligned with the X axis of force sensor as shown in Figure 4.12 in order to be tracked by the camera. The OpenCV library is used to detect the checkerboard pattern, similar to the calibration procedure performed for the Kinect. Having the transformation from the checkerboard coordinate frame to camera coordinate frame, one can obtain the overall transformation from the probe sphere tip frame to camera frame, since the transformation from the board frame to the sensor and the tip frames are constant. The following overall transformation is used to compute the probe sphere tip position.

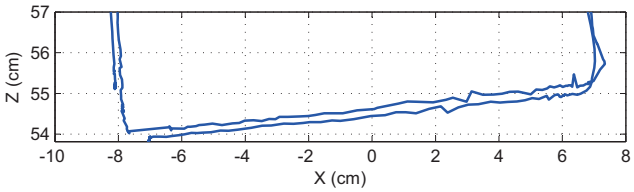
$$\boldsymbol{T_{Tip \rightarrow Sensor}} = \boldsymbol{T_{Chessboard}}\ \boldsymbol{T_{Sensor}}\ \boldsymbol{T_{Tip}} \tag{4.20}$$

OpenCV `findChessboardCorners` function extracts the 2D corner points of the checkerboard attached to the handle. As mentioned before, this function gives the location of the corners on the image plane. The 2D projection points and the 3D object points along with camera matrix and distortion coefficients are used by OpenCV `solvepnpransac` method to estimate the pose of the checkerboard. The function `solvepnransac` finds such a pose that minimizes re-projection error, i.e., the sum of squared distances between the observed image points by `findChessboardCorners` function and the projected object points using camera parameters. A re-projection error threshold is also fed into that function which is based on the RANSAC algorithm [22]. It matches the largest number of inlier feature points on the coplanar target with its image.

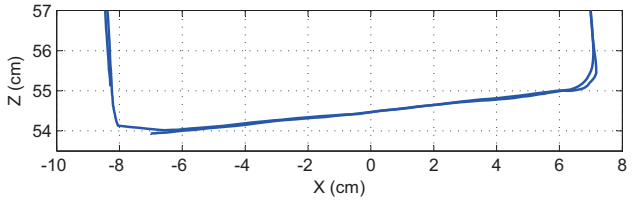
Filtering

The Kalman filter algorithm [33] optimizes the stream of noisy data from a sensor output. It is used for estimating the position of the probe, since raw data from the checkerboard pose estimates are noisy due to hand movement, illumination variation, or other environmental factors.

The goal of the Kalman filter equation is to find an estimate $\hat{x}_k$ for the current state of the system. This estimate is based on the previous estimate $\hat{x}_k^{'}$ and a weighted difference of an



(a) Before Kalman filter



(b) After Kalman filter

Figure 4.13: Part of the CD case tracking before and after applying Kalman filter.

actual measurement z_k and a measured prediction $H\hat{x}_k^{'}$ as shown in Equation 4.21.

$$\hat{x}_k = \hat{x}_k^{'} + K_k(z_k - H\hat{x}_k^{'}) \tag{4.21}$$

$$K_k = \frac{P_k^{'}H^T}{(HP_k^{'}H^T + R)}$$

$$P_k = (1 - K_k^{'})P_k^{'} + 1$$

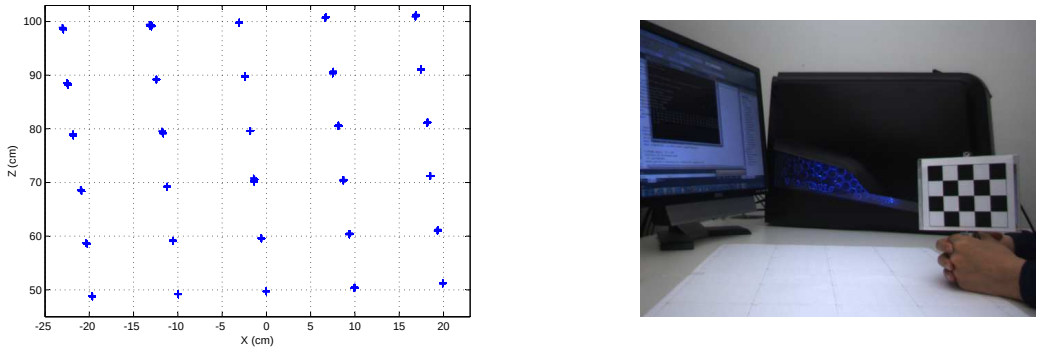
where K is the optimal Kalman gain, H is the observation model, P is the estimate error covariance, R is measurement error covariance, and index k shows the state or measurement at time interval k. The final state estimate takes into account both measurement estimates from sensor and the estimates from the model. In our experiment the H and R are static scalars for all time intervals and the values are set as follows:

$$H = 1,\ R = 10,\ P_0 = 0$$

Figure 4.13 shows the tracking result before and after Kalman filter applied.

4.2.3 Workspace Error

In order to measure the tracking deviation in the workspace, a grid of 5×6 points is prepared with distance of 10 cm between grid lines that provides a workspace of 40×50 cm. The grid



(a) Workspace points (b) Workspace

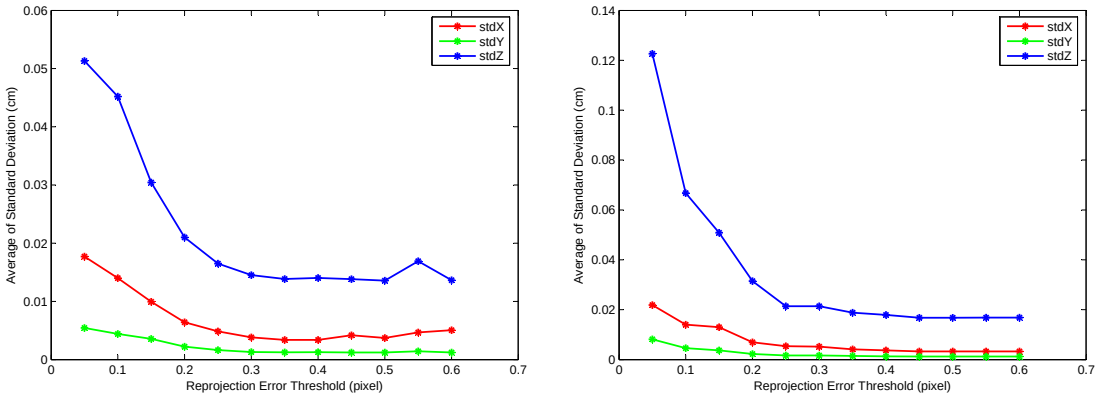
Figure 4.14: Workspace sampling.

is placed in 50 *cm* distance from the camera and the sensor tip is positioned on each point of the grid once at a time to capture 100 frames per point. As shown in Figure 4.14 the data is captured for 30 points in total. Data captured in this experiment is used both for determination of the re-projection error threshold for each camera, and error analysis.

Re-projection error threshold which is one of the input parameters to `solvepnpRansac` function, is the maximum allowed distance in pixels between the observed and computed point projections to be considered as an inlier. The goal is to find the best re-projection error threshold for the Bumblebee camera and the Kinect to be used when tracking checkerboard markers. Regarding this, the workspace captured data is tracked using re-projection error threshold ranging from 0.05 to 0.6 with steps of 0.05. For each threshold value, first, the standard deviation is calculated over 100 captures of a point, then, the average deviation over the workspace is computed for the 30 points.

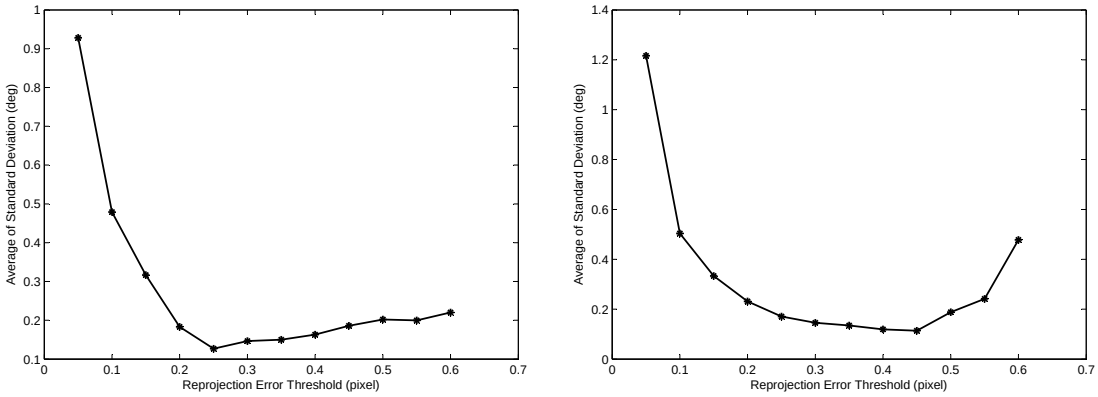
$$\underset{j=1:30}{\mu}\left(\underset{i=1:100}{\sigma}(x)\right),\ \underset{j=1:30}{\mu}\left(\underset{i=1:100}{\sigma}(y)\right),\ \underset{j=1:30}{\mu}\left(\underset{i=1:100}{\sigma}(z)\right),\ \underset{j=1:30}{\mu}\left(\underset{i=1:100}{\sigma}(\alpha)\right) \tag{4.22}$$

where $\sigma(x)$ means standard deviation of stationery sensor tip position in direction of x axis over 100 frames which is calculated per grid point. Then, the average of this deviation is computed over all the grid points in the workspace. Moreover, the same calculation is performed



(a) Bumblebee (b) Kinect

Figure 4.15: Average of standard deviation of workspace in x, y, and z directions.

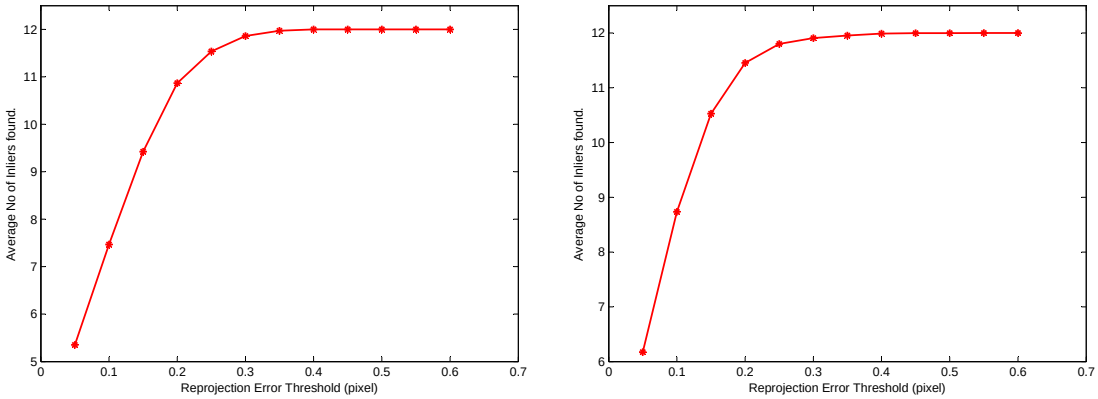


(a) Bumblebee (b) Kinect

Figure 4.16: Average of standard deviation of angle between the normal of the checkerboard and Z axis of camera.

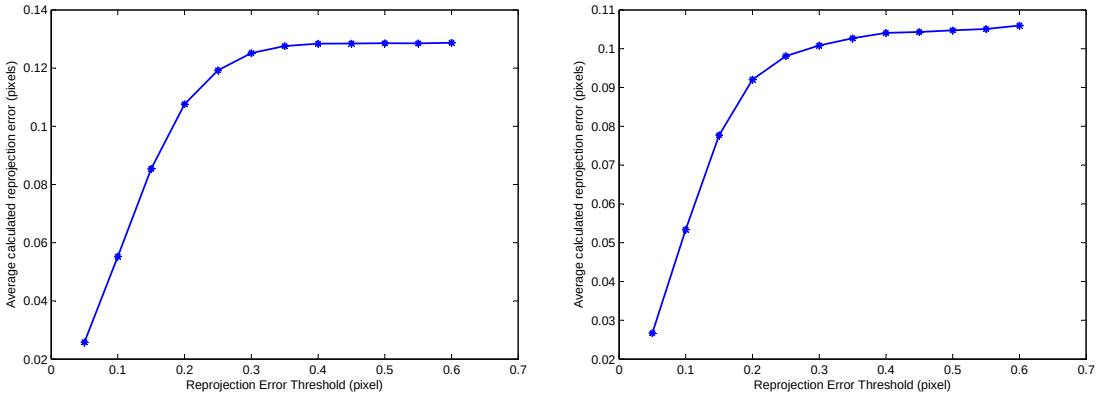
for the angle between the normal of the target and z axis of the camera (α). Figure 4.15 shows the average deviation of the sensor tip position in the x, y, and z direction detected by the two cameras. It can be seen that a lower threshold values results in more deviation for stationery points due to the small number of feature points detected as inliers. Figure 4.16 shows the average deviation of angle α for different thresholds. It clearly demonstrates that the threshold value resulting in the least deviation for the cameras are 0.25 for Bumblebee, and 0.45 for the Kinect. If a higher threshold value is selected, it will mislead the inlier selection.

Figure 4.17 illustrates the average number of inliers selected by solveRansac function over



(a) Bumblebee (b) Kinect

Figure 4.17: Average of inliers over workspace.

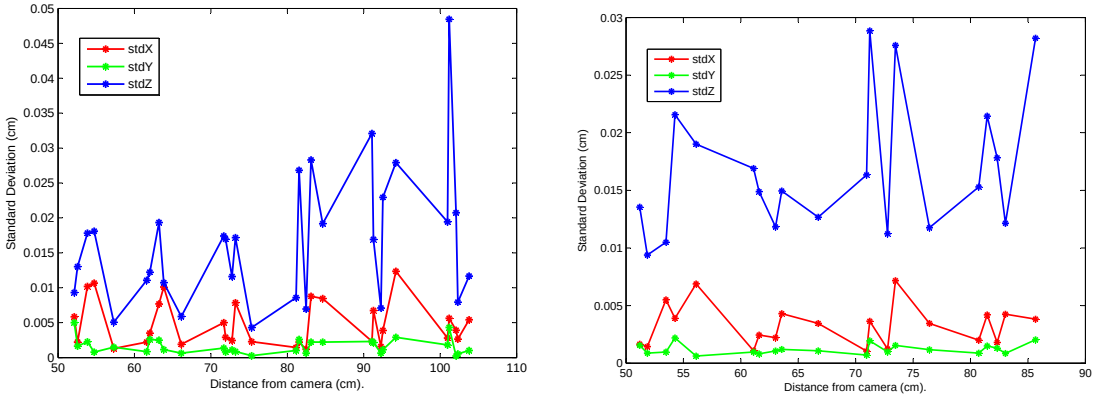


(a) Bumblebee (b) Kinect

Figure 4.18: Average of calculated reprojection error over workspace.

workspace grid points and 100 frames capture $\left(\underset{j=1:30}{\mu}\left(\underset{i=1:100}{\mu}(inlier)\right)\right)$. As shown in the figure, the maximum number of feature points to be selected as inliers for the marker is 12. Although increasing the threshold guarantees finding the total number of feature points but it also increases the calculated re-projection error, which is the sum of the squared distance between 2D image points (obtained from corners extraction) and projected 2D points using the current pose estimation. Figure 4.18 shows the average amount of estimated re-projection error over frames and workspace points $\left(\underset{j=1:30}{\mu}\left(\underset{i=1:100}{\mu}\left(\hat{E}_R\right)\right)\right)$.

Having the re-projection error threshold determined, the standard deviation of the sensor tip



(a) Bumblebee (b) Kinect

Figure 4.19: Standard deviation in workspace.

position can be calculated over the workspace area. It is illustrated in Figure 4.19 as a function of grid point distance from the camera. Table 4.2 also shows the average of the deviations in 3 directions for both cameras. As expected the deviation in the Z axis direction is more than in the other directions. It should be noted that the workspace area for the Kinect is $40 \times 30\ cm$, because the marker is not detected at a further distance due to lower resolution of the Kinect images. In summary, test was carried out to give an estimate of the error in the workspace and to allow the selection of the threshold for the re-projection error used for the visual tracking function.

(cm)	Bumblebee	Kinect
$\underset{j=1:30}{\mu}\left(\underset{i=1:100}{\sigma}(x)\right)$	0.0048	0.0033
$\underset{j=1:30}{\mu}\left(\underset{i=1:100}{\sigma}(y)\right)$	0.0016	0.0012
$\underset{j=1:30}{\mu}\left(\underset{i=1:100}{\sigma}(z)\right)$	0.0165	0.0168

Table 4.2: Average of standard deviation of sensor tip position over workspace for the selected re-projection error threshold.

4.2.4 Probe Trajectory test

A CD case trajectory test is carried out to check the accordance of the visual tracking with the captured point cloud. The CD case contour point cloud was measured to be a rectangular box

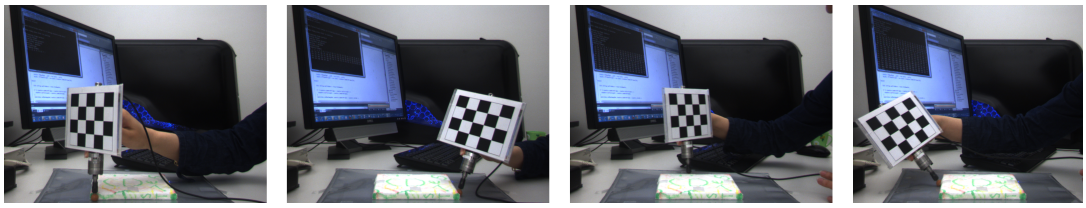
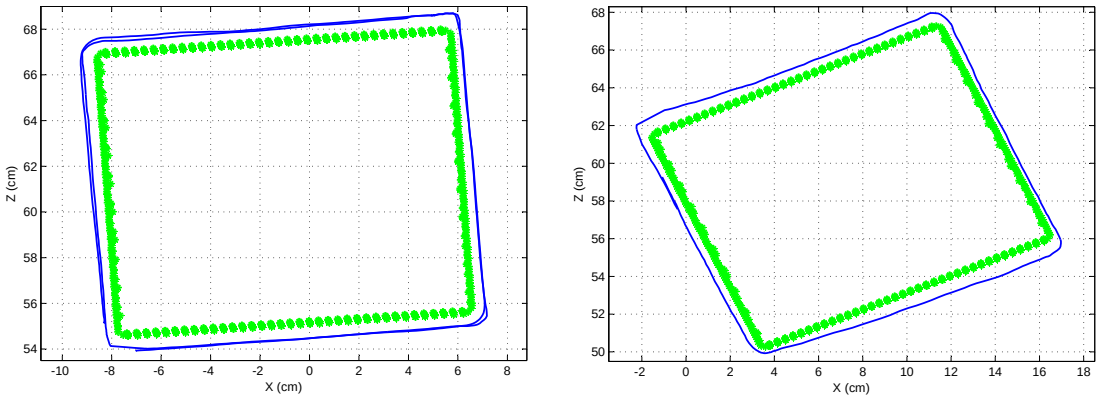


Figure 4.20: Trajectory test around a CD case.



(a) Bumblebee (b) Kinect

Figure 4.21: Trajectory obtained from tracking in blue against the point cloud in green.

of 14.2×12.4 *cm* in the two major dimension. The box was then registered to the captured point cloud using Meshlab. The outline of the case was traced over a planar surface using the force sensor as shown in Figure 4.20. Figure 4.21 shows the tracing curves have an offset from the points due to the sensor tip radius which is 6.22 *mm*. Since the Bumblebee camera has higher resolution and exact calibration parameters, its tracking result is more precise. Some errors due to inaccuracies caused by tracing by hand are inevitable.

4.3 Surface Tracking

One of the requirements of the parameter estimation is to have the complete mesh deformation, in order to compute the displacement from the initial configuration. We use the surface tracking method presented and implemented by Wuhrer et al. [75]. My contributions to this paper was data acquisition and pre-processing. The contributions to the algorithm design and the implementation were performed as part of this thesis but S. Wuhrer and J. Lang are the main

authors. The goal of this approach is to track the complete geometry of a deforming object using the point cloud data captured from a single point of view. The main feature of this algorithm is that it is robust and tolerant to noisy point cloud data. This algorithm is divided into two main steps: First, a tracking method is used to deform a complete mesh model. Second, using the result of the first step for observed vertices in a FEM, the position of the unobserved vertices are estimated and improved. An overview of the algorithm is given in this section.

The inputs of this method are a closed template mesh, a contact point, a force direction, and a series of deformation frames in a point cloud format. The template mesh should be aligned to the first frame before the tracking process starts. The goal of the tracking method is to compute a deformed template mesh for each captured frame. An energy minimization approach is employed to find the transformation matrix for each vertex of the template mesh as shown in the following relation:

$$\boldsymbol{A_i} = \boldsymbol{A_{trans}}(\boldsymbol{p_i})\boldsymbol{A_{rot}}(\boldsymbol{r_i}, \phi_i)\boldsymbol{A_{trans}}(\boldsymbol{t_i})\boldsymbol{A_{trans}}(-\boldsymbol{p_i}) \tag{4.23}$$

where $\boldsymbol{A_i}$ is the transformation matrix corresponding to vertex $\boldsymbol{p_i}$ in the template mesh. $\boldsymbol{A_i}$ depends on 3 parameters to build the translation $\boldsymbol{t_i}$, two parameters to build the rotation axis $\boldsymbol{r_i}$, and a parameter for rotation angle ϕ. The two transformations $\boldsymbol{A(p_i)}$ and $\boldsymbol{A(-p_i)}$ are to center the coordinate frame at vertex $\boldsymbol{p_i}$, so that the difference between the neighbor vertices can be computed directly. After estimating the 6 parameters which build the transformation matrix $\boldsymbol{A_i}$, the position of the unobserved points are improved using the known displacement of observed points, the contact point, and force direction by means of a FEM. Finally, the transformation matrix $\boldsymbol{A_i}$ is updated to take the new position of unobserved points into account.

4.3.1 Tracking the observed vertices

The goal of this section is to deform the template mesh to the observed point cloud data. An energy minimization approach is performed to estimate the transformation parameters required to build the transformation matrix $\boldsymbol{A_i}$. The first step is to consider the whole template mesh

as a rigid object and transform it to fit the first frame input data. For rigid alignment step, the distance between template vertices and their nearest neighbor point on the point cloud data is minimized using the following equation:

$$E_{initial} = \sum_i w_i \left\| \boldsymbol{A}\tilde{\boldsymbol{p}}_{\boldsymbol{i}} - \boldsymbol{N}(\boldsymbol{p}_{\boldsymbol{i}}) \right\|_2^2 \tag{4.24}$$

$$w_i = \begin{cases} 1 & \alpha < \alpha_{thresh},\ \left\| \boldsymbol{A}\tilde{\boldsymbol{p}}_{\boldsymbol{i}} - \boldsymbol{N}(\boldsymbol{p}_{\boldsymbol{i}}) \right\|_2 < dr \\ 0 & otherwise \end{cases}$$

where $\boldsymbol{A}$ is the global rigid transformation, $\tilde{\boldsymbol{p}}_{\boldsymbol{i}}$ is the homogeneous coordinate of the template vertex, and $\boldsymbol{N}(\boldsymbol{p}_{\boldsymbol{i}})$ is its nearest neighbor in the point cloud data. The weight term w_i is one if the angle between the normal of $\boldsymbol{p}_{\boldsymbol{i}}$ on template mesh and the corresponding point $\boldsymbol{N}(\boldsymbol{p}_{\boldsymbol{i}})$ in the point cloud is less than a threshold while the distance between these corresponding points is also less than dr. In order to avoid the misalignment of template mesh vertices near the boundary of the point cloud, their associated w_i are set to zero. A point is considered to be on the boundary of the point cloud when chosen by many vertices of the template mesh.

The next step is to align the template to the point cloud data in a non-rigid way. In other words, estimate the transformation parameters $\boldsymbol{t}_{\boldsymbol{i}}$, $\boldsymbol{r}_{\boldsymbol{i}}$, ϕ_i by minimizing the tracking energy:

$$E_{track} = w_{data} E_{data} + w_{sm} E_{sm} \tag{4.25}$$

$$E_{data} = \sum_i w_i \left\| \boldsymbol{A}_{\boldsymbol{i}}\tilde{\boldsymbol{p}}_{\boldsymbol{i}} - \boldsymbol{N}(\boldsymbol{p}_{\boldsymbol{i}}) \right\|_2^2$$

$$E_{sm} = \sum_i \frac{1}{|R_{sm}(p_i)|} \sum_{j \in R_{sm}(p_i)} \left(\left\| \boldsymbol{t}_{\boldsymbol{i}} - \boldsymbol{t}_{\boldsymbol{j}} \right\|_2^2 + min\left(\left(\phi_i - \phi_j\right)^2, \left(2\pi - |\phi_i - \phi_j|\right)^2 \right) \right)$$

where w_{data} and w_{sm} are weights of each energy terms, $R_{sm}(p_i)$ is the set of indices corresponding to points $\boldsymbol{A}_{\boldsymbol{j}}\tilde{\boldsymbol{p}}_{\boldsymbol{j}}$ which are points within a geodesic distance of $s_{sm}r$ from $\boldsymbol{A}_{\boldsymbol{i}}\tilde{\boldsymbol{p}}_{\boldsymbol{i}}$. s_{sm} is a parameter and r is the average edge length of the template mesh. The tracking energy consists of two parts: the first part carries template mesh closer to input data by minimizing

the distance of corresponding points in template mesh and point cloud frame; and the second part smooths the mesh by keeping the rotation and translation similar for neighboring vertices. The data term of the energy uses only 3 equations to estimate 6 parameters per observed vertex which results in an underconstrained problem. Therefore, the smoothness term is added to apply more constraints on the system which causes a smooth transformation.

In the beginning of the optimization, w_{sm} and w_{data} are typically set to 100 and 1 respectively. As we proceed, the smoothness weight is halved in each iteration $w_{sm} = w_{sm}/2$ to give more weight to the data term. Finally, the optimization stops if w_{sm} is smaller than 20 or the relative change in energy is less than 0.0001 ($\left(E_{track}^{(i-1)} - E_{track}^{(i)}\right) / E_{track}^{(i-1)}$ where i is the number of iterations).

The amount of mesh deformation depends on the template mesh resolution because the resolution limits the distance from a template mesh vertex to its nearest neighbor in the data energy term. In order to allow larger deformation, a multi-resolution approach is used in this tracking method. A multi-resolution hierarchy of the template mesh is obtained by eliminating edges of the mesh halving the number of vertices in each resolution level. Each edge elimination is performed only if the resulting mesh is not self-intersecting. This process stops when either no valid edge for elimination is found or the mesh has around 1000 vertices.

The optimization in multi-resolution mode operates as follows. The transformation parameters for the lowest resolution mesh (resolution level l) are assigned to either identity ($\boldsymbol{t_i}$ and $\boldsymbol{r_i}$ are null vectors, ϕ_i is zero) or to the result of the previous frame. Then, E_{track} is minimized for the mesh of resolution level l. The next step is to set the transformation parameters of the higher resolution mesh (resolution level $l+1$). For the vertices present in level l, the parameters are set to the result of level l. Then, energy E_{sm} is minimized to solve for the parameters of the remaining vertices only. Once the parameters are set for all vertices, E_{track} is minimized as before.

As a result of this optimization, we have a set of transformation parameters for each frame that aligns the template mesh to that frame. Since we use single view point cloud data for frames, there are vertices in the template mesh that are not associated to a valid input data.These vertices are called unobserved vertices and were deformed by smoothness term in optimization. The transformation for these vertices are updated using FEM which is explained in Section 4.3.2.

First we have to define which vertices are observed. Lets define B_i as the number of template mesh vertices that chose $\boldsymbol{N(p_i)}$ in the data points as nearest neighbor. $\boldsymbol{N(p_i)}$ represents the data point in the point cloud corresponding to at least a vertex on the template mesh like $\boldsymbol{p_i}$. Then, the average count is $B_{ave} = \sum B_i/n$, where n is the number of data points that were chosen at least by one vertex of the template mesh.

The observed vertices have to satisfy two conditions: first, their associated point $\boldsymbol{N(p_i)}$ in the point cloud has a corresponding weight $w_i = 1$; second, the corresponding count B_i is less than twice the average count ($B_i < 2B_{ave}$). This is to prevent the poor assignment of vertices to data points, in which a vertex correspondent to a hole in data is assigned to wrong data point.

4.3.2 Updating unobserved vertices using FEM

The goal of this section is to improve the position of unobserved vertices in the template mesh. Using a linear FEM that relates the displacement of vertices to the force applied to them, helps updating the position of unobserved vertices. In this step a tetrahedral mesh is created from the simplified template mesh to compute the FEM. For each frame the tetrahedral mesh vertex position is updated by computing the displacement between the current and the previous frame surface mesh. Equation 4.26 shows that the force vector applied to tetrahedral mesh $\boldsymbol{f}$ is related to displacement vector $\boldsymbol{u}$ using a stiffness matrix $\boldsymbol{K}(E, \nu)$ that depends on the geometry of the tetrahedral mesh and two elasticity parameters, Young's modulus and Poisson's ratio.

$$\boldsymbol{K}(E, \upsilon)\boldsymbol{u} = \boldsymbol{f} \tag{4.26}$$

Having known the displacement for observed vertices, the contact point and direction of force at the contact point and zero force at all the remaining vertices except support vertices, we could find the displacement of the unobserved points, if we know the elasticity parameters. Therefore, we first minimize Equation 4.27 with respect to E and ν, using only points with known forces and displacements. Once the elasticity parameters are estimated, we use this information to compute the displacement of unobserved vertices.

$$(\boldsymbol{K}(E, \upsilon)\boldsymbol{u} - \boldsymbol{f})^2 \tag{4.27}$$

It should be noted that FEM step is used for the purpose of regularization and does not have physical meaning since it is performed between each consecutive frame and not between the first and current frame. Algorithm 3 shows the procedure of estimating the displacement of unobserved vertices. As a result of the tracking step, a smooth deformed template mesh is obtained. As the first step, the displacement per vertex is computed between current and previous frame for surface vertices leaving the interior node displacements unknown, but amongst the surface displacements only the displacement associated to observed vertices are reliable. The displacements of the interior vertices are computed using thin-plate spline (TPS) deformation based on Daryden and Mardia [21]. To this end, all the forces and displacements are known except for unobserved vertices which have unreliable displacements and support surface vertices which have unknown forces. An iterative method is used to update the support surface forces as follows: E and ν are estimated minimizing Equation 4.27, the estimated parameters are then used to build matrix $\boldsymbol{K}$ and solve for unknown forces using Equation 4.26. The iteration continues until the support surface force values are converged. In the final step, the known forces, estimated parameters, and observed vertices displacements are used to find the unobserved vertices displacements.

The final step is to adjust the transformation parameters associated to unobserved points

Algorithm 3 Using FEM to displace unobserved vertices

Data: $T^{tet}_{F_{j-1}}$ (previous frame solid mesh), F_j (current frame surface mesh)

Result: $\boldsymbol{u}$ (unobserved vertices displacement)

begin

Compute $\boldsymbol{u_{init}}$ based on F_j and a thin-plate spline deformation

Initialize a set F of indices corresponding to known forces;

for l iterations **do**

Use the known $\boldsymbol{f_i}$ to solve for E, ν by minimizing $\|F = \boldsymbol{K}(E, \upsilon)\boldsymbol{u} - \boldsymbol{f}\|^2$

Solve for the unknown $\boldsymbol{f_i}$ using $\boldsymbol{K}\boldsymbol{u}_{init} = \boldsymbol{f}$

Update F to contain all estimated forces

end for

Use the estimated $E, \nu, \boldsymbol{f}$ and observed $\boldsymbol{u}$ to find $\boldsymbol{u}$ at unobserved vertices

end

to consider the new deformation. This is achieved by minimizing the following equation:

$$E_{def} = w_{data}E_{target} + w_{sm}E_{sm} \tag{4.28}$$

$$E_{target} = \sum_i \|\boldsymbol{A_i}\tilde{\boldsymbol{p}}_{\boldsymbol{i}} - \boldsymbol{TP}(\boldsymbol{p_i})\|_2^2$$

where $\boldsymbol{TP}(\boldsymbol{p_i})$ is the position of corresponding point $\boldsymbol{p_i}$ in tetrahedral mesh. It should be noted that in this minimization step only the transformation parameters of unobserved vertices are modified.

4.4 Summary

In this chapter the building blocks employed to setup the measurement and computation of input data to the parameter estimation system were presented. The devices used for data acquisition were introduced, their working range and mechanism were briefly explained, including the principal of the stereo camera and the Kinect and the calibration procedure. As data preparation step, the background removal procedure was explained, and the force sensor tip was located using visual tracking. Some tests are then performed to make sure the data

is synchronized, and the error in probe tracking is under control. Finally, the surface tracking method which is used for the full mesh deformation was explained.

Chapter 5

Results

In this chapter the results obtained from three test objects are presented and discussed. In Section 5.1 the Hertzian contact model is briefly described and is applied to data recorded during indentation of a foam block. Section 5.2 presents and compares results obtained from applying displacement and force-displacement error minimization, respectively, for all test objects. Finally, Section 5.3 summarizes the results.

5.1 Hertzian Contact Model

In contact mechanics a simple model to describe the deformations that occur for elastic bodies of basic geometry is the Hertzian contact. It applies to two spheres in general, or a sphere and a plane in particular, that slightly deform under an applied force. This theory, considers the effect of geometry on local elastic deformation, such that the circular contact area of a sphere and a plane are related to the elastic properties of the materials. It omits surface interactions such as contact adhesive interaction, and assumes the two surfaces are fully elastic.

In an experiment of indenting a foam block the force sensor indenter is a sphere and the foam block is large relative to the sphere and planar, a simple parameter estimation is performed using the Hertzian contact model [38]. The foam block material is soft, forces are relatively small, the indentation occurs quasi-static and we assume, fully elastic deformation is assumed.

Since the indenter is a grinding stone, it is assumed to be infinitely stiff relative to the foam block. Therefore the Hertzian contact model [38] for a sphere-plane contact is

$$F = \frac{4}{3} E^* R^{\frac{1}{2}} u^{\frac{3}{2}}, \quad E^* = \frac{E}{1-\nu^2} \tag{5.1}$$

where F is the applied force, R is the indenter radius, u is indented depth, E^* is reduced Young's modulus, E is Young's modulus, and ν is Poisson's ratio.

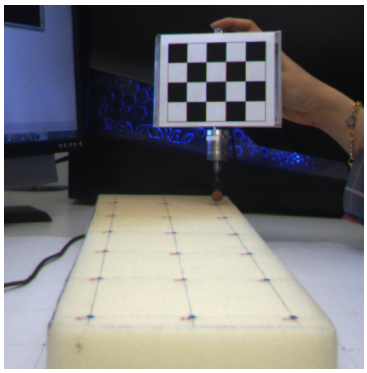
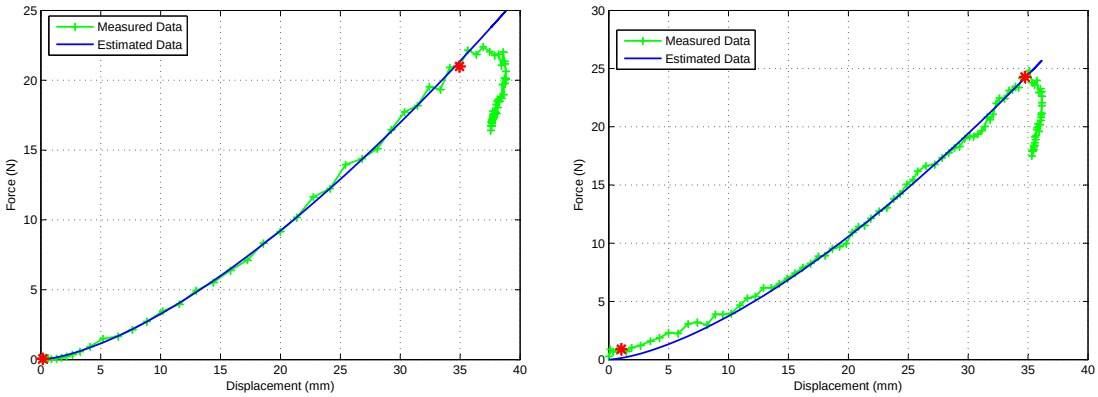


Figure 5.1: Sampling of grid points on foam block.

As the first test object, an open cell foam block is selected with the dimension of $598 \times 150 \times 47.5$ mm in length, width, and height respectively. A grid of points is drawn on top of the foam block, and a dense sampling is carried out on 18 points with 10 cm distance from each other (Figure 5.1). For each indentation test, the applied force and the sensor tip location is recorded and used to compute the depth of indentation. Figure 5.2 shows the Hertzian contact model that is fit to the data points with a nonlinear minimization using Matlab's fminsearch. The model curve is shown in blue, while the recorded data is in green. The deviation of the model from the data points is due to the viscoelastic effect of the object which is neglected in the modeling. Red points in the figure indicate the range of data used for estimation. The sample where the force readings start to differ from zero is selected as lower bound, and the upper bound is where the deformation stops and the force starts to drop because of the viscoelastic effect which we do not model. The measured curve overestimates the force response due to elastic deformation because of the non-zero velocity of the indenter during the deformation.



(a) Bumblebee (b) Kinect

Figure 5.2: Fitting Hertzian contact model with data recorded by Bumblebee and Kinect. This figure is associated to region number 8, the same is done for all the other regions.

From the curve fitting the stiffness coefficient $\frac{4}{3}E^*R^{\frac{1}{2}}$ is estimated and Equation 5.1 is then employed to compute E^*. The same procedure is performed for each grid point on the foam block. Figure A.1 and A.2 in the appendix, show an acceptable curve fitting to measured data for all the grid points. It confirms the consistency of the model throughout the foam block. In order to demonstrate the consistency of data capture for a single point, the measurement is repeated 16 times for point number 11 on the foam block. Figure 5.3 shows the repeatedly measured data in green, which is aligned based on where the force magnitude is greater than 0.3 N. A model is fitted to each curve and the reduced Young's Modulus is estimated which is shown in Table 5.1. The mean and standard deviation of the computed E^* is 30800 and $2700N/m^2$ respectively. Table 5.2 shows the reduced Young's modulus on each point of the foam block. These values are represented by a color map in Figure 5.4 with region number indicated on each block. The values are within the range of 10000 to $30000N/m^2$ and show that the foam is stiffer in the central regions compared to borders of the material, which is expected because of the violation of the infinite plane assumption of the Hertzian contact model.

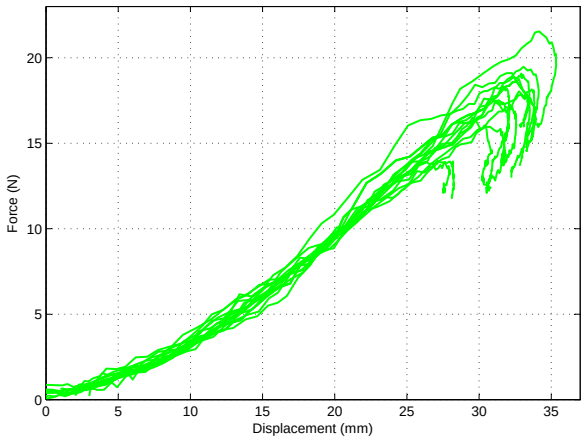


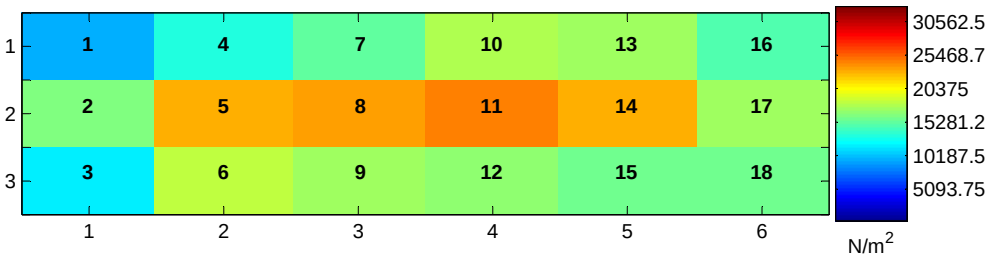
Figure 5.3: Repeated recordings by Kinect on sample point 11.

Attempt	1	2	3	4	5	6	7	8
E^*	29400	26600	29900	32600	34100	28200	31700	30900
Attempt	9	10	11	12	13	14	15	16
E^*	32800	36000	30700	33300	32300	28900	30100	26000

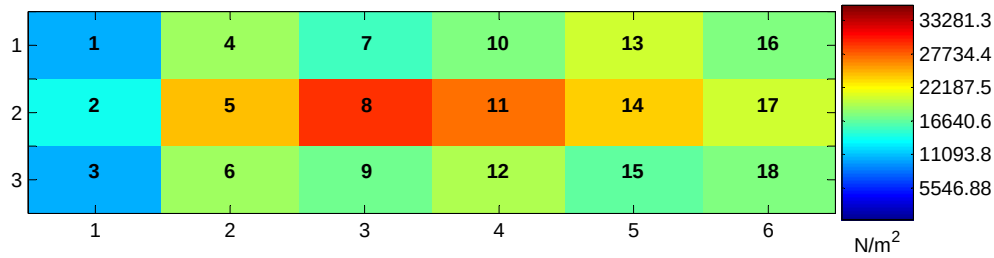
Table 5.1: Repeatedly measured data recorded from indenting point 11 on the foam block.

5.2 Displacement error minimization

The displacement error minimization is explained in Chapter 3, and applies the Nelder-Mead simplex algorithm to estimate the elastic parameters. In this section results of tests with three objects with differing behavior are presented using the proposed approach. First, a foam block is tested and the results are compared to the Hertzian model for the same object. Second, the associated result of a compressed foam ball is presented. Finally, a silicon anatomical ear phantom with more complex geometry is tested and the results are presented. The foam block is selected because of its simple geometry and planarity so that both the Hertzian contact model and the Displacement error minimization method could be tested on the same object and compared with each other. The foam block is not very stiff and exhibits only local deformation. Thus another simple geometry object was selected, a ball, that deforms more globally and it is stiffer compared to the foam block. The ear model which is between the foam block and foam ball in terms of stiffness, is a more challenging object geometrically. In each of these tests, two



(a) Bumblebee



(b) Kinect

Figure 5.4: A color map from top view of foam block showing the stiffness modulus of each region. The labels on the axis represent the row and column of grid point on width and length of foam block respectively.

Bumblebee	N/m^2	1	2	3	4	5	6
	1	13100	17600	20500	23400	22900	19800
	2	21900	30300	31100	32600	30300	22900
	3	15400	24400	22900	22600	21100	20700
Kinect	N/m^2	1	2	3	4	5	6
	1	12800	22900	19100	21400	25300	21800
	2	16700	29800	35500	32700	29300	25000
	3	13100	22900	20900	23400	20300	21700

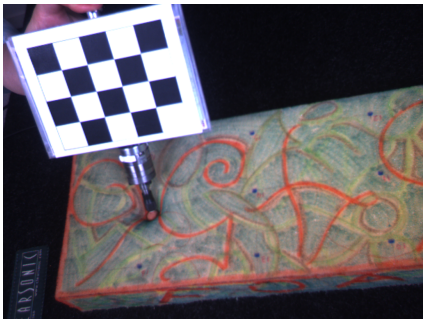
Table 5.2: E^* estimated by applying Hertzian contact model for grid points on foam block. The row and column labels represent the row and column of grid point on width and length of foam block respectively.

options for error calculation: displacement error and force-displacement error as explained in Chapter 3, are considered and discussed.

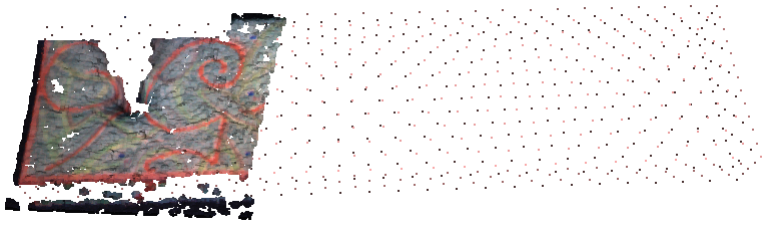
The implementation is developed in C++ and executed on a Dell machine with Intel Core i7 CPU, 12GB RAM, and a 64-bit Windows 7 operating system. Open source libraries have been used in parameter estimation implementation including tetgen for mesh tetrahedralization, lapack for solving linear least square, and Nelder-Mead Algorithm for optimization.

5.2.1 Foam Block

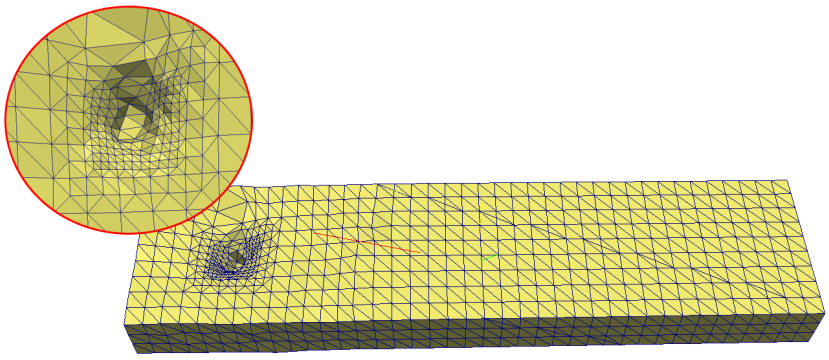
When deforming the foam block manually, one can observe that it deforms locally and almost no global bulging occurs. In order to obtain visually correct results from surface tracking, extra points are added to the captured point cloud representing the part of the foam block that is not observed but remains undeformed. The extra points help rigid alignment of the template with the point cloud for each frame. Figure 5.5b shows the captured point cloud along with extra points added to it. These points help to provide constraints on the side of the foam block in which no captured points exist, otherwise the captured points could be placed anywhere along the length of the foam block. Figure 5.6 shows one possible output of rigid alignment failure. The captured point cloud is associated with grid point 17, which is located on the far left end of the template mesh, but due to lack of constraint points on the sides it slid towards the middle of the template mesh.



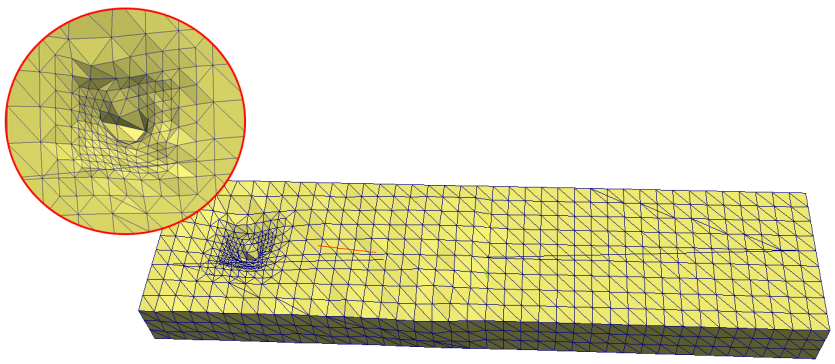
(a) Color image



(b) Point cloud



(c) Bumblebee tracking result



(d) Kinect tracking result

Figure 5.5: Foam block deformation color, point cloud recordings, and the tracking results.

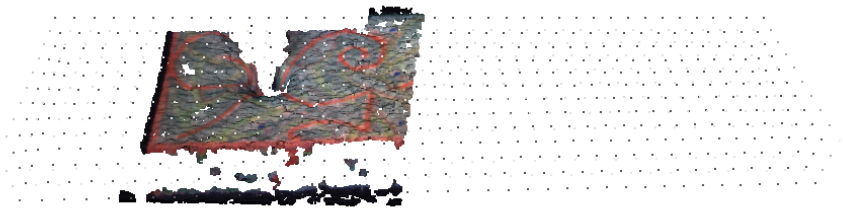


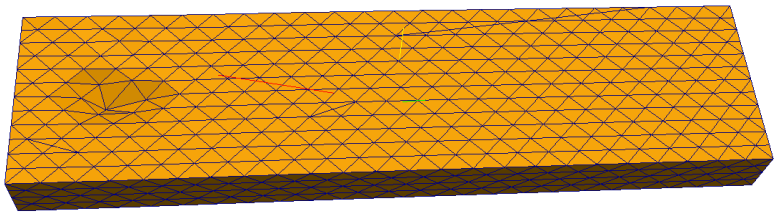
Figure 5.6: Rigid alignment is failed due to lack of data points on the sides of foam block. The captured point cloud showing deformation of sample point 17 instead of sitting on far left of the template mesh (black disperse points), is slid in the middle of it because it has no side constraints.

The other modification from Wuhrer et al. [75] to obtain the correct tracking result is that the template mesh is refined around the contact point to improve tracking of highly localized deformations as shown in Figure 5.5c and 5.5d. Furthermore, some extra points on a hemisphere are added to the captured point cloud to represent the lower part of the sphere indenter in touch with the surface of the foam block. The position of the center of the sphere and the orientation of the sensor probe are known from visual tracking section. These data along with the radius of the sphere allow the location of the hemisphere. These modifications enable to more realistically deform the mesh appearing close to the actual observations. Figure 5.7 shows the tracking result which looks wider and shallower without the modifications as shown in Figure 5.5.

The tracking results for two different sensors, Bumblebee and Kinect, are shown in Figure 5.5. There are two main differences between the tracking results. First, the contact point for the Kinect does not look as correct as the Bumblebee. This is likely because the distance of the foam block from the Kinect had to be further than using the Bumblebee, thus the Kinect was not able to capture the local deformation clearly. Second, the surface of the foam block captured by the Kinect is slightly rough due to some glitter effect of the foam block material. Therefore, the Bumblebee tracking produces better results for foam block.

The input mesh used for tracking and parameter estimation consists of 1201 vertices. Furthermore, the parameters used for tracking include the data weight, smoothness weight and

(a) Side view



(b) Top view

Figure 5.7: Foam block deformation without re-meshing the contact area. The pictures show side view and top view of the slightly deformed object.

smoothness radius were set to $w_{data} = 1$, $w_{sm} = 10$, and $s_{sm} = 3$ respectively and the optimization terminates if w_{sm} is smaller than 1.

The displacement error minimization method is applied on three grid points on the foam block and for each point both error options in the minimization (combined force-displacement error, and displacement error only), are tested. The optimization is run for 100 iterations. Figure 5.8 illustrates the change of the material parameters λ and μ as well as the error. However, as can be seen from the plot after 40 iterations the result does not change any further. Similar optimization behavior can be observed with the two other objects: the ball and the ear. Therefore, only the iterations up to the point where the minimization does not effect the estimated parameters is shown in the remaining results.

Figure 5.9 shows the result before convergence for sample point number 16 on foam block, considering the combined force-displacement error. The plot for λ and μ demonstrates larger

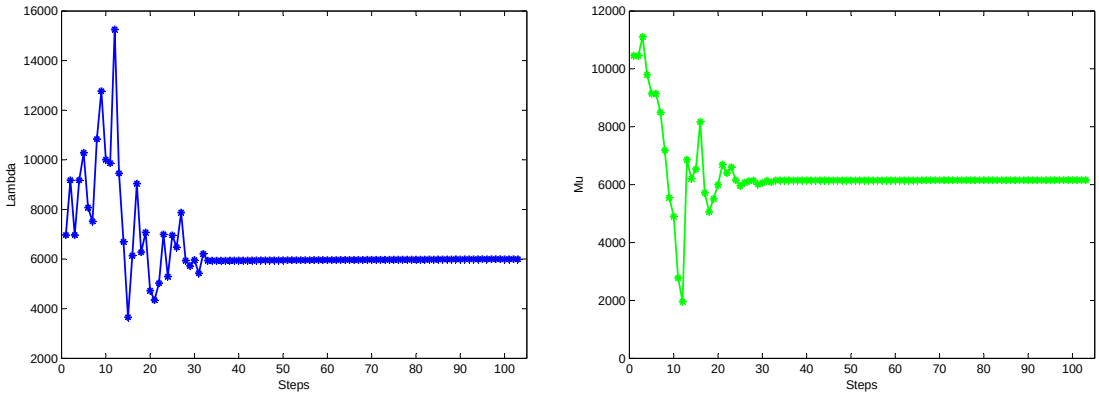
oscillations in the first 20 iterations, then it gradually decreases until convergence. The average force-displacement error per vertex declines to 2 mm from 2.2 mm.

The Lamé parameters can be converted to Young's modulus and Poisson's ratio to obtain an approximate relationship to stiffness and volume preservation (see Section 3.1.3). The equivalent Young's modulus and Poisson's ratio is expressed in Table 5.3. It shows reasonable Poisson's ratio compared to observation, while Young's modulus is underestimated compared to the result from fitting the Hertzian model. However, it follows the pattern of stiffer sample points in the middle of the foam block with sample point 16 having the lowest stiffness seen previously in Section 5.1.

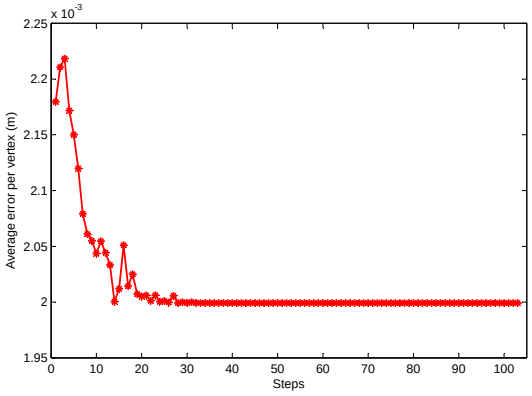
The force-displacement error option results in lower Young's modulus compared to the Hertzian model, therefore another option of error calculation is also tried considering only displacement error. Table 5.3 shows the corresponding results. The estimated Young's modulus is closer to the Young's modulus obtained with the Hertzian model fitting, but the Poisson's ratio has also increased compared to using the combined force-displacement error.

Figure 5.10 demonstrates the displacement mismatch between input deformation which is the result of surface tracking, and the computed deformation using linear FEM. There are some spots on the top of the foam block with higher error which are indicated by lighter color, especially for the tracking result based on the Bumblebee data. It should be noted that noise in the surface tracking output has an effect on these mismatches.

The results obtained from data captured by the two sensors are similar, however it has higher estimated values for data captured by the Kinect. Although the Bumblebee camera has a higher resolution and is more precise compared to the Kinect, it will not capture point clouds from objects with no visual features. This is the reason for adding some color features to the foam block so that it could be acquired. Whereas, the Kinect uses active illumination and does not have such issue.

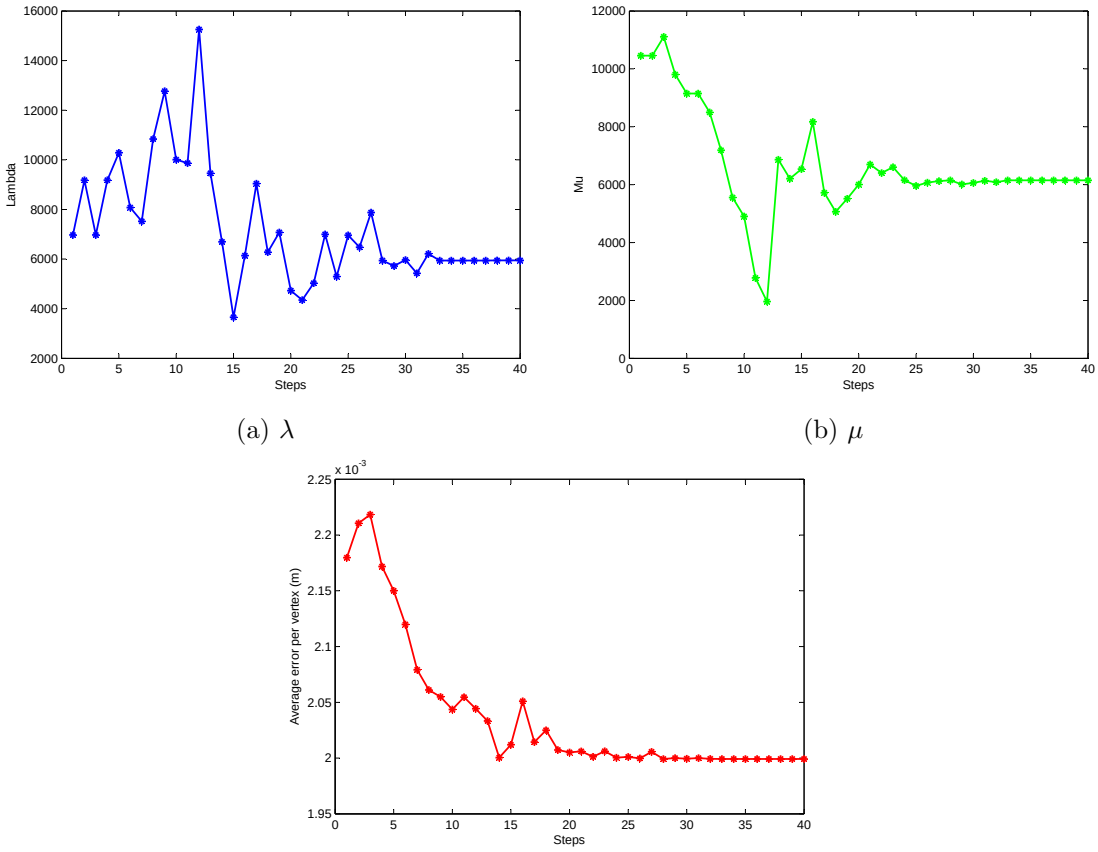


(a) λ (b) μ



(c) Displacement and Force Error

Figure 5.8: Parameter optimization of point number 16 on foam block considering both force and displacement error.

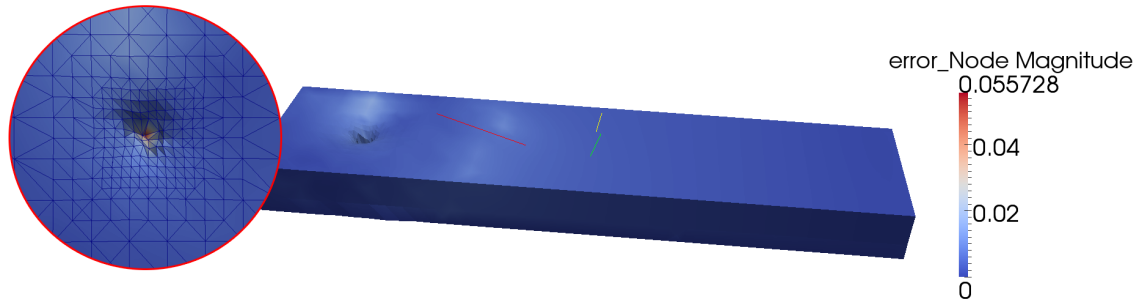


(a) λ

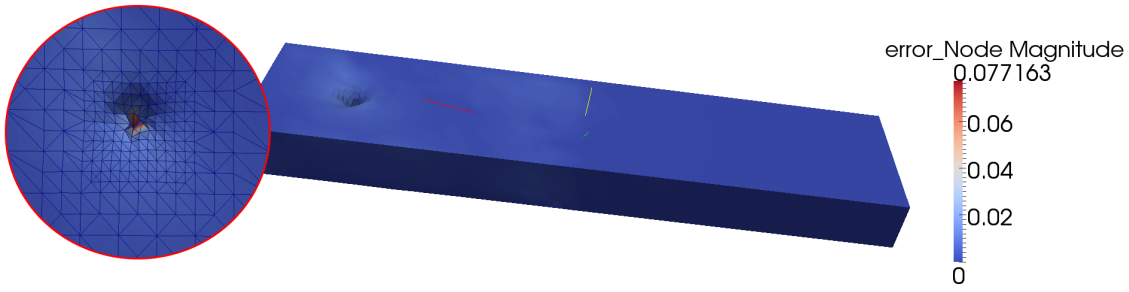
(b) μ

(c) Displacement and Force Error

Figure 5.9: Parameter optimization of point number 16 on foam block considering both force and displacement error (first 40 iterations).



(a) Bumblebee



(b) Kinect

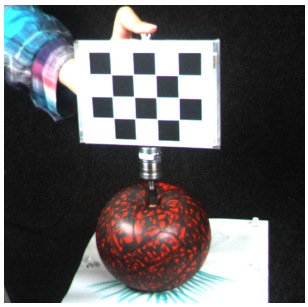
Figure 5.10: The mismatch between measured deformation and computed deformation using linear FEM. The magnitude of nodal error is presented in meters.

Bumblebee	Force-Displacement Error			Displacement Error		
Foam block point	No. 8	No. 16	No. 17	No. 8	No. 16	No. 17
E	19165	15357	15574	24100	20163	24406
ν	0.3460	0.2467	0.2559	0.4410	0.3998	0.3120
E^*	21771	16352	16665	29919	23999	27038
E^* (Hertzian)	31100	19800	22900	31100	19800	22900
Kinect	Force-Displacement Error			Displacement Error		
Foam block point	No. 8	No. 16	No. 17	No. 8	No. 16	No. 17
E	19997	19173	19432	35499	31756	34507
ν	0.3023	0.1415	0.2785	0.3120	0.1681	0.2650
E^*	22008	19565	21066	39327	32679	37113
E^* (Hertzian)	35500	21800	25000	35500	21800	25000

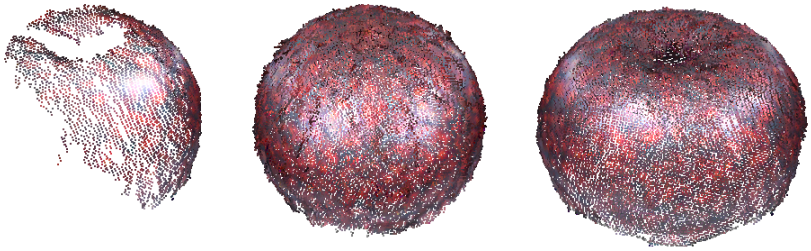
Table 5.3: Estimated parameters for 3 points on the foam block using error minimization method.

5.2.2 Ball

The second tested object for displacement error minimization method is a closed cell foam ball with the radius of 61.4 mm with a smooth patternless surface. Comparing to foam block, it has stiffer material which requires greater force to be applied for the same indentation, but the indentation results in more global bulging. An issue with the ball is that it does not have a geometrical features, which results in attracting the template mesh vertices towards the single view data points during the tracking. This means the whole mesh shrinks and the volume is not preserved. In order to address this issue, the single view point cloud is rotated such that the supporting surface points on the base of the ball are used as alignment points to form a symmetrically deformed object, as it is actually appears. To reconstruct the full view of the point cloud, some extra points on the support surface are added to the single view point cloud which is in the form of a hexagon. The point cloud is then replicated six times by aligning the hexagon on the base of the ball such that each time the point cloud is rotated to another side of the hexagon. The alignment is accomplished with the use of Meshlab. In Figure 5.11 and 5.13 the single view and rotated point cloud along with the associated tracking result are shown. As it can be seen when using the single view point cloud in Figure 5.11, the tracking result is a ball squished towards the data points which is shown in Figure 5.13a. However, when the augmented point clouds shown in the middle and right of Figure 5.11b is fed into the surface



(a) Color image



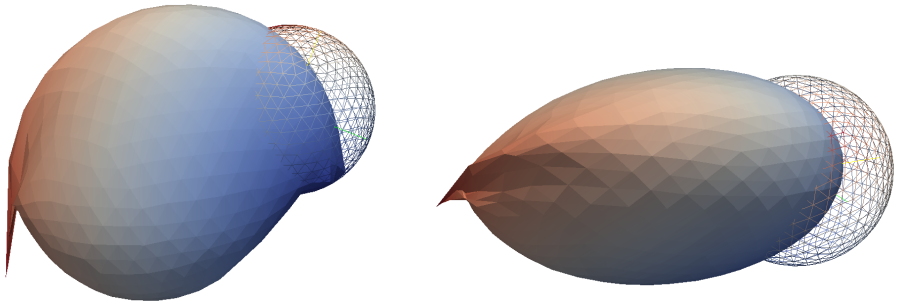
(b) Point cloud

Figure 5.11: Ball point cloud. On left is single view point cloud, in middle is rotated undeformed point cloud, and on right is rotated deformed point cloud.

tracking, the output is more plausible.

A modification on the contact nodal force which is mentioned in Section 3.3.1 and is performed for all the test objects, is that the measured force is rotated to the direction of the contact triangle in order to make the deformation more realistic. Otherwise, the deformation looks like Figure 5.12. The whole mesh is compressed on one side, grown on the other side, and tilted. The wire frame mesh represents the initial configuration of the ball to compare with the resulting mesh. This is caused by shear forces and depends on the mesh resolution, the support surface size, and the amount of deformation. This effect is more clear for the ball since its mesh has lower resolution, smaller support surface, and larger deformation.

Figure 5.13b and 5.13c shows the tracking result obtained from data captured by the Bumblebee and the Kinect. Since the Bumblebee is at a closer distance and captures the local deformation more precisely around the contact point, more realistic deformation can be ob-

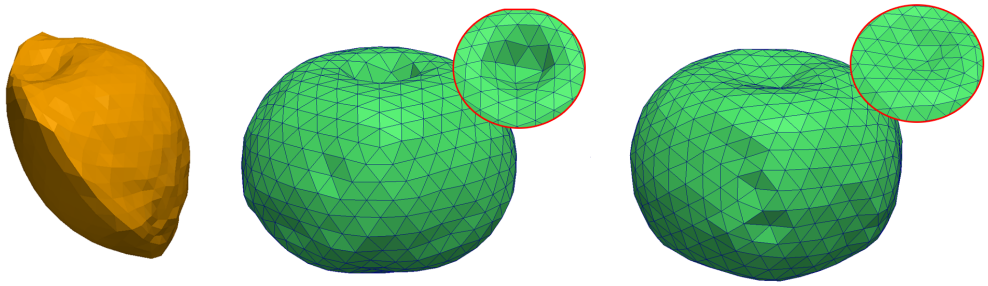


(a) Side view (b) Top view

Figure 5.12: The effect of original force direction on the ball.

served in the tracking result compared to the Kinect. However, both of the results show the global bulging based on the captured point clouds. In fact, the Kinect captured better global point clouds for this object. The Bumblebee was not able to capture the regions with specular highlights (Figure 5.11a) caused by the material of the ball.

The input mesh used for tracking and parameter estimation consists of 642 vertices. The parameters used for tracking the foam block are also used for tracking the ball. It includes $w_{data} = 1$, $w_{sm} = 10$, $s_{sm} = 3$, and the termination condition of w_{sm} is 1. A modification applied to tracking the ball is again (as for the foamblock) the hemisphere of extra points added to the point clouds representing the sphere indenter.



(a) Single view (b) Bumblebee (c) Kinect

Figure 5.13: Ball tracking result. On the left is the result when using a single view point cloud, and in 5.13b and 5.13c are the results using the rotationally augmented point clouds captured by the two sensors.

As with foam block, the optimization is carried out for 100 iterations, but the result of the first 55 iterations is shown in Figure 5.14. The plots for λ and μ show constant rise at the beginning, and some oscillation when the optimization is closer to convergence. The average error per vertex is around 6 mm after convergence form 19 mm which is higher than for the foam block, due to the more global deformation.

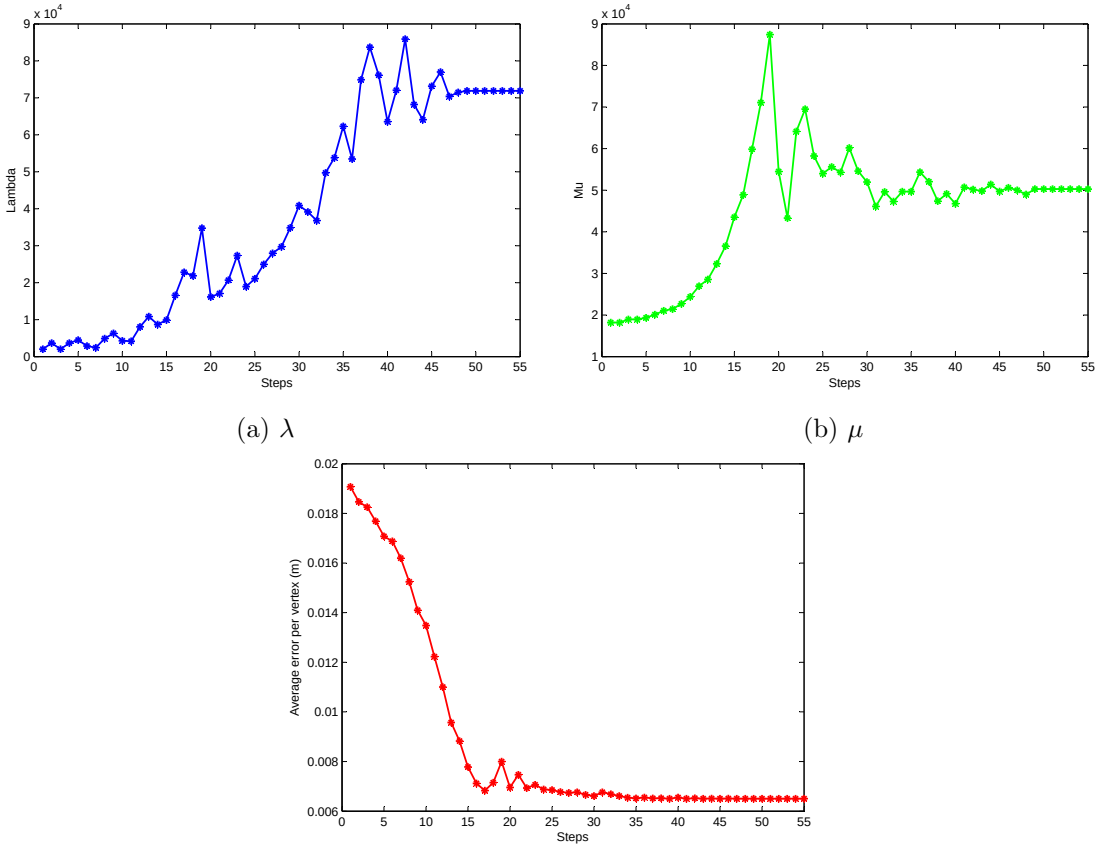
As with the foam block, the equivalent Young's modulus and Poisson's ratio are presented in Table 5.4, showing both options of error calculation. The error in the results are consistent with the foam block results. In other words, the option of displacement error only, yields higher Young's modulus and Poisson's ratio.

Figure 5.15 demonstrates the deformation mismatch between computed and input displacement. In both of the figures, there is approximately few millimeters of nodal mismatch. This could be explained because of the large deformation of the ball, and the errors of the surface tracking results due to the need to rotationally complete the point clouds.

The parameters estimated using Kinect data once again have higher values for Young's modulus compared to Bumblebee data, but the Poisson's ratio is almost the same. The two main issues in capturing point clouds of the foam ball, the lack of visual features and the shininess of the material, made it hard for the Bumblebee, while the Kinect captured very clean point clouds of the object.

	Force-Displacement Error		Displacement Error	
	Bumblebee	Kinect	Bumblebee	Kinect
E	130160	145490	325370	372530
ν	0.2942	0.2975	0.4980	0.4180

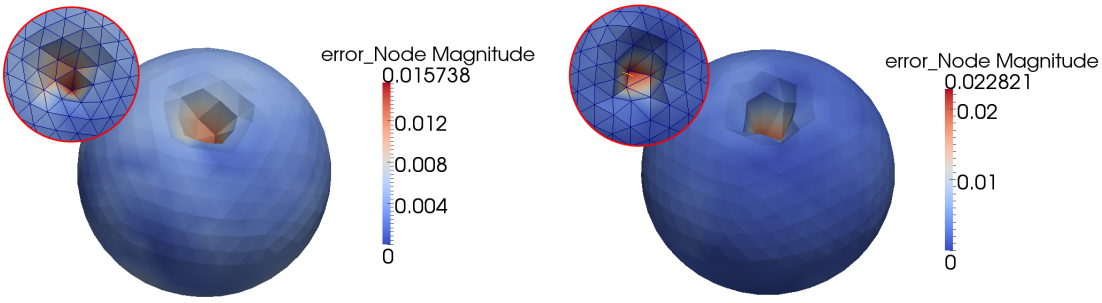
Table 5.4: Estimated parameters for the foam ball using two different error minimization method and data captured with two different sensors.



(a) λ (b) μ

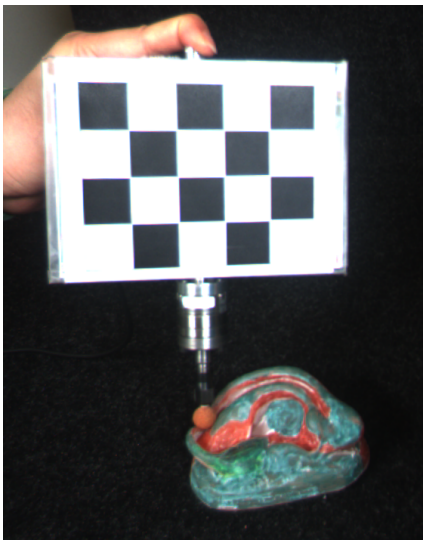
(c) Displacement and Force Error

Figure 5.14: Parameter optimization of foam ball considering both force and displacement error.

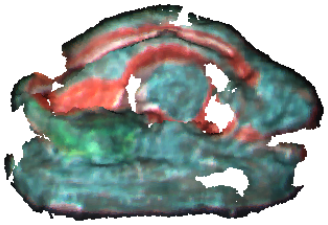


(a) Bumblebee (b) Kinect

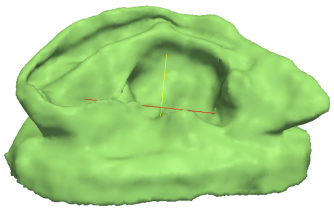
Figure 5.15: The mismatch between measured deformation and computed deformation using linear FEM. The magnitude of nodal error is presented in meters.



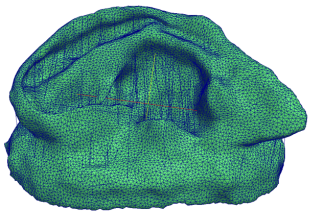
(a) Color image



(b) Point cloud



(c) Tracking result



(d) Result with edges

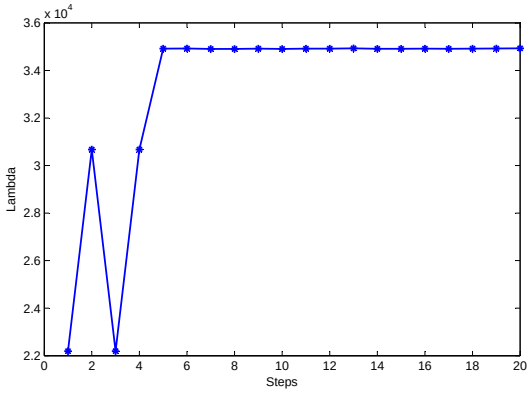
Figure 5.16: Ear model deformation.

5.2.3 Ear

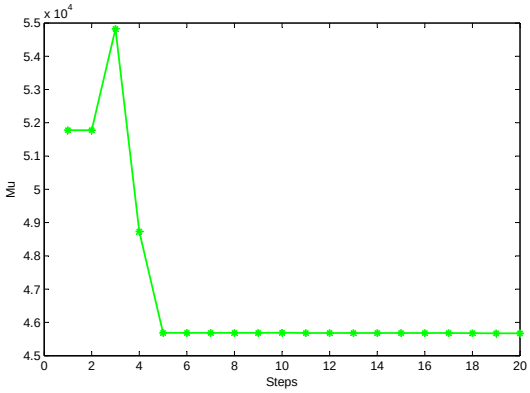
As a final experiment, the method is examined on a more geometrically complex silicon anatomical ear phantom. The bounding box of the ear model has the dimension of $94 \times 62 \times 38$ mm in length, width, and height, respectively. Comparing to the other two objects tested, the ear model was more compliant than the ball but stiffer than the foam block, therefore required a force magnitude between what was applied to the other two objects to deform. The object is more locally deformed and is not much affected globally. Since the model is relatively small it could only be captured by the Bumblebee, since the Kinect requires at least 50 cm distance from the object to capture the point cloud. Some visual patterns are also added to the featureless model, so that the Bumblebee camera is able to capture the point clouds. Figure 5.16 shows the color and point cloud of last frame of deformation process along with its tracking result. The displacement of the object is very small due to its size, and the only part deforming is the earlobe. In the deformation process the upper part of earlobe is pressed downwards and the Bumblebee is setup such that it captures large point clouds covering both, the deformed and undeformed part of the object.

The mesh employed for tracking has 19993 vertices, while the one used for FEM part of the tracking and parameter estimation has 2993 vertices. Unlike the two other objects that used a modified version of tracking procedure, the ear model used the tracking of Wuhrer et al. [75] without any modification. The parameters used in tracking include $w_{data} = 1$ and $w_{sm} = 100$ which are the weight of the data and smoothness energy terms, and $s_{sm} = 1.5$ for the smoothness radius. The termination condition for w_{sm} is 20.

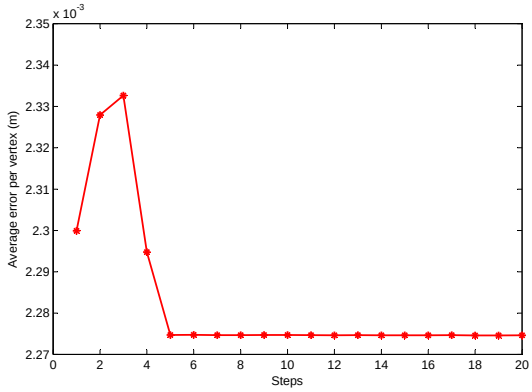
Figure 5.17 demonstrates that the optimization converges very quickly. This behavior may be explained by the small deformation of the object, and does not show oscillations around the converged value. The average displacement error per vertex is 2.27 mm from 2.3 mm.



(a) λ



(b) μ



(c) Displacement and Force Error

Figure 5.17: Parameter optimization of silicon ear considering both force and displacement error.

Table 5.5 shows the equivalent Young's modulus and Poisson's ratio for the two error measures. Once again, considering only displacement error results in higher Young's modulus and Poisson's ratio. Furthermore, Figure 5.18 shows the mismatch between computed deformation and the tracking result. The higher error occurs on the part that has global deformation in the tracking result, but in the linear FEM simulation mainly local deformation occur. This may indicate some global motion present in the tracking result.

	Force-Displacement Error	Displacement Error
E	99022	111010
ν	0.2170	0.3575

Table 5.5: Estimated parameters for ear model using error minimization method, considering both option error calculation.

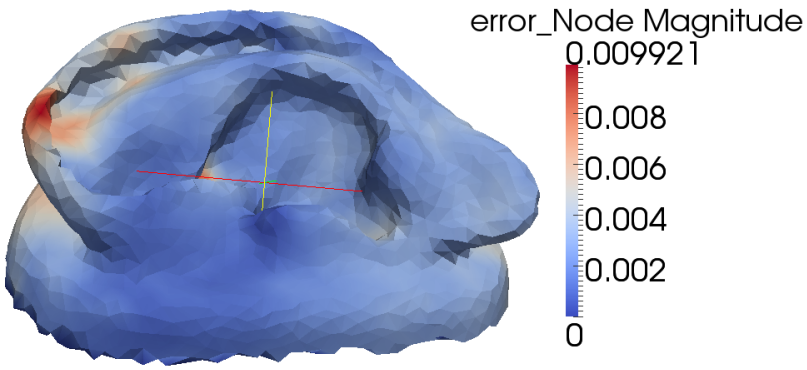


Figure 5.18: The mismatch between measured deformation and computed deformation using linear FEM. The magnitude of nodal error is presented in meters.

5.3 Summary

In this chapter the results for three tested objects were presented, applying the Hertzian contact model and displacement and/or force error minimization. Initially, the Hertzian contact model was applied to a grid of sample points on a foam block. The stiffness modulus was estimated for each sample point, and the pattern showed higher stiffness on the center points which was as expected. Next, the displacement error minimization method was tested on three objects: foam block, foam ball, and rubber ear model.This error minimization method was investigated using two options as error measures: displacement error, and force-displacement error. For the foam block, three sample points were tested and the results were compared to the results of Hertzian contact model. The force-displacement error option underestimated the Young's modulus, while the displacement error option had results closer to the Hertzian contact model. In fact, the parameters resulting from the other two objects confirm that the displacement error option yields higher Young's modulus and Poisson's ratio compared to the force-displacement error option. Amongst these three objects, the foam ball could be observed to preserve its volume, which results in the estimation of higher Poisson's ratios compared to the other two objects. Moreover, each set of data was captured twice, once with the Bumblebee camera and another time with the Kinect, except for ear model. Generally, for higher resolution, more precise data, and smaller objects the Bumblebee camera is the better choice. However, if the

object is featureless, or has a material that reflects light, the Kinect may be preferable.

Chapter 6

Conclusion

In this chapter a summary of the thesis is presented, and some improvements proposed as future work.

6.1 Summary

In this thesis, we have presented two approaches for elasticity parameter estimation based on 3D point cloud data and force measurements. We have outlined the steps to generate a complete deformed mesh from captured point cloud data and synchronized force measurements. We have addressed some data pre-processing steps such as background removal, and visual tracking and positioning of the force sensor probe.

We based our approach on the linear FEM parameter estimation proposed by Becker and Teschner [3]. The difference between their method and ours is that we also perform a forward FEM as part of our method. In this approach the linear relation between applied force and resulting displacement is re-arranged such that the elasticity parameters can be retrieved using linear least squares. One of the issues when discretizing the problem is determining the nodal force and displacements. The surface node displacements are computed using the difference between the initial frame and the current complete frame mesh, which is obtained from the surface tracking of Wuhrer et al. [75]. The displacements may contain errors from the tracking

result. The internal node displacement remains part of the problem that we solve. In addition to the surface tracking, we assume an external force applied to a single triangle of the mesh and is not distributed in the neighbor nodes. The applied force to contact triangle is not the exact measured force, but the contact area and the direction of the triangle normal is also taken into account. Apart from contact nodes, other nodes have zero force applied to them except for the support surface nodes which remain with unknown forces in the problem. Therefore, some nodal displacements and forces are part of the problem unknowns along with the two elasticity parameters.

The first approach is to optimize the problem in a two step method: once solving for unknown nodal force and displacements, then solving for elasticity parameters. We call this approach the linear method. It was tested on an input data obtained from a FEM simulation. The method's success depends on the exact values from the simulation result and is not tolerant to any error or noise in the nodal values. Therefore a second approach is developed that minimizes the displacement and/or force errors using Nelder-Mead simplex algorithm. The second approach gave more reasonable results for the FEM simulation data, compared to the first approach.

Finally, point cloud data and force measurements were recorded for three test objects. The first object is a foam block with a grid of points drawn on it. Data is recorded while indenting each of these points. At first, the Hertzian contact model is used to estimate an approximate of stiffness for each grid point. Then the displacement error minimization method is used to estimate the two elasticity parameters for three grid points. This method is also used to estimate the parameters of a foam ball and a silicon ear model. In this method two options for error calculation were used: displacement error, and force-displacement error. The results demonstrated that larger parameters were estimated using the displacement error option compared to the force-displacement error. The data was also captured by two depth sensors: the Bumblebee and the Kinect with the data from the Kinect resulting in larger parameters. We found that the results for displacement error option based on the Bumblebee data is closer to the Hertzian

contact model results compared to the other error option and the results for the Kinect.

6.2 Future work

For the future work, we are interested in improving some simplifications performed in this work. One improvement is to try the corotational FEM model instead of linear FEM, in order to allow for larger deformations and avoid artifacts from the linear FEM. The improvement in the computational method has an impact on the shear forces that cause the unnatural growth of the object in larger deformations.

Another improvement is to consider more contact nodes depending on the resolution of the mesh and employing an interpolation model to gradually decrease the force magnitude applied to mesh nodes as their distance from the contact point increases. This modification improves the local deformation visually and reduces the force error since the assigned forces to the mesh would be more realistic.

In addition, our method can be improved by making it compatible to the input data from different devices. Of particular interest are imaging modalities used for elastography. The output data from elastography imaging can be processed and the required information extracted such that our method can be used to estimate the parameters of internal organs.

Appendix A

Hertzian contact model fitting for foam block

This appendix contains the Hertzian contact model curve fitting to measured data for all the grid points on the foam block. The numbers for each plot indicate the sample point number shown in Figure 5.4. For more detail see Section 5.1.

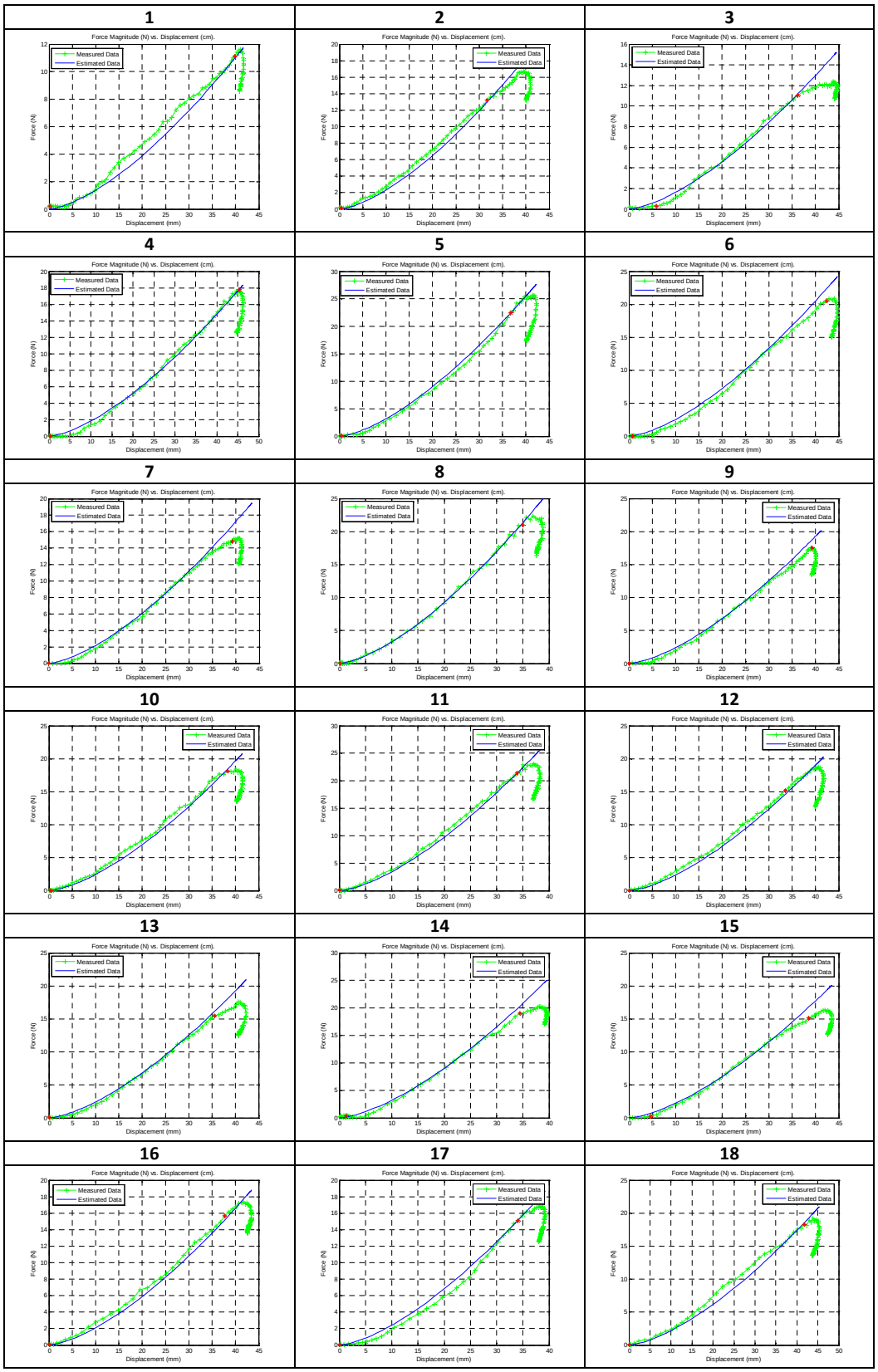


Figure A.1: Fitting Hertzian contact model to measured data from sample point grid on foam block recorded by Bumblebee

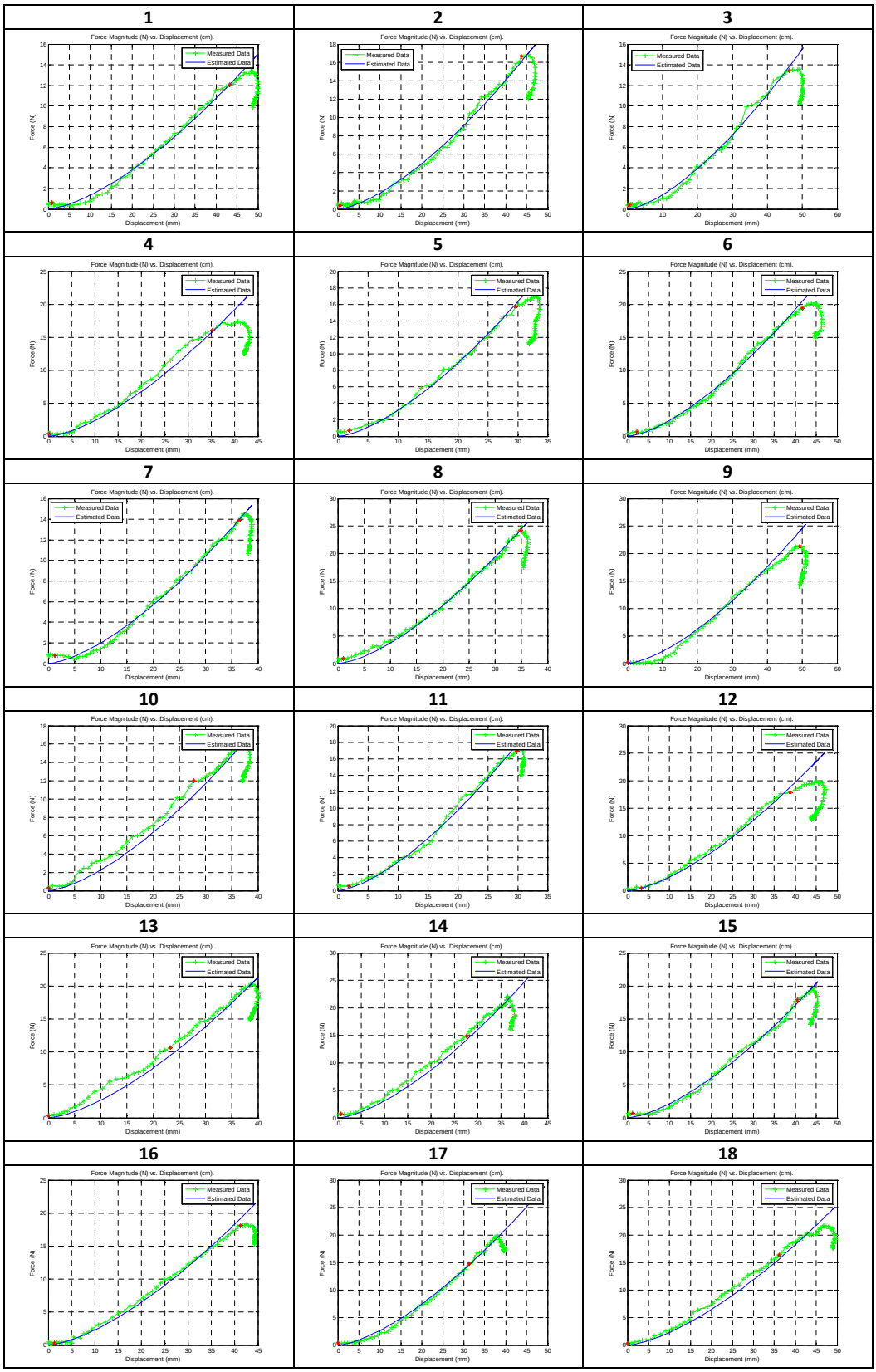


Figure A.2: Fitting Hertzian contact model to measured data from sample point grid on foam block recorded by Kinect.

Bibliography

[1] Bummo Ahn and Jung Kim. Measurement and characterization of soft tissue behavior with surface deformation and force response under large deformations. *Medical image analysis*, 14(2):138, 2010.

[2] Klaus-Jürgen Bathe. *Finite element procedures*, volume 2. Prentice hall Englewood Cliffs, 1996.

[3] Markus Becker and Matthias Teschner. Robust and efficient estimation of elasticity parameters using the linear finite element method. *Proc. of Simulation and Visualization*, pages 15–28, 2007.

[4] Nicolás Benech and Carlos A Negreira. Longitudinal and lateral low frequency head wave analysis in soft media. *The Journal of the Acoustical Society of America*, 117:3424, 2005.

[5] Kiran S Bhat, Christopher D Twigg, Jessica K Hodgins, Pradeep K Khosla, Zoran Popović, and Steven M Seitz. Estimating cloth simulation parameters from video. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 37–51. Eurographics Association, 2003.

[6] Gérald Bianchi, Barbara Solenthaler, Gábor Székely, and Matthias Harders. Simultaneous topology and stiffness identification for mass-spring models based on fem reference deformations. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2004*, pages 293–301. Springer, 2004.

[7] Bernd Bickel, Moritz Bächer, Miguel A Otaduy, Wojciech Matusik, Hanspeter Pfister, and Markus Gross. Capture and modeling of non-linear heterogeneous soft tissue. In *ACM Transactions on Graphics (TOG)*, volume 28, page 89. ACM, 2009.

[8] Pasu Boonvisut, Russell Jackson, and M Cenk Cavusoglu. Estimation of soft tissue mechanical parameters from robotic manipulation data. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4667–4674. IEEE, 2012.

[9] Morten Bro-Nielsen. *MEDICAL IMAGE REGISTRATION AND SURGERY SIMULATION IMM-PHD-1996-25*. PhD thesis, Technical University of Denmark, 1997.

[10] Morten Bro-Nielsen. Finite element modeling in surgery simulation. *Proceedings of the IEEE*, 86(3):490–503, 1998.

[11] Morten Bro-Nielsen and Stephane Cotin. Real-time volumetric deformable models for surgery simulation using finite elements and condensation. In *Computer graphics forum*, volume 15, pages 57–66. Wiley Online Library, 1996.

[12] Steve Burion, Francois Conti, Anna Petrovskaya, Charles Baur, and Oussama Khatib. Identifying physical properties of deformable objects by using particle filters. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1112–1117. IEEE, 2008.

[13] David T Chen and David Zeltzer. *Pump it up: Computer animation of a biomechanically based model of muscle using the finite element method*, volume 26. ACM, 1992.

[14] Eric J Chen, Jan Novakofski, W Kenneth Jenkins, and William D O'Brien Jr. Young's modulus measurements of soft tissues with application to elasticity imaging. *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on*, 43(1):191–194, 1996.

[15] APC Choi and YP Zheng. Estimation of young's modulus and poisson's ratio of soft tissue from indentation using two different-sized indentors: finite element analysis of the finite

deformation effect. *Medical and Biological Engineering and Computing*, 43(2):258–264, 2005.

[16] Timothy R Coles, Dwight Meglan, and Nigel W John. The role of haptics in medical training simulators: a survey of the state of the art. *Haptics, IEEE Transactions on*, 4(1):51–66, 2011.

[17] Stéphane Cotin, Hervé Delingette, and Nicholas Ayache. Real-time elastic deformations of soft tissues for surgery simulation. *Visualization and Computer Graphics, IEEE Transactions on*, 5(1):62–73, 1999.

[18] Brett R Cowan, Ian J LeGrice, Poul MF Nielsen, and Alistair A Young. Method and apparatus for soft tissue material parameter estimation using tissue tagged magnetic resonance imaging. 2005.

[19] A-M Cretu, Emil M Petriu, and Pierre Payeur. Data acquisition and modeling of 3d deformable objects using neural networks. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 3383–3388. IEEE, 2009.

[20] MM Doyley. Model-based elastography: a survey of approaches to the inverse elasticity problem. *Physics in medicine and biology*, 57(3):R35, 2012.

[21] IL Dryden and KV Mardia. *Statistical analysis of shape*. Wiley, 1998.

[22] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[23] Barbara Frank, Markus Becker, Cyrill Stachniss, Wolfram Burgard, and Matthias Teschner. Learning cost functions for mobile robot navigation in environments with deformable objects.

[24] Barbara Frank, Rüdiger Schmedding, Cyrill Stachniss, Matthias Teschner, and Wolfram Burgard. Learning the elasticity parameters of deformable objects with a manipulation robot. In *Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[25] Barak Freedman, Alexander Shpunt, Meir Machline, and Yoel Arieli. Depth mapping using projected patterns, April 2 2008. US Patent App. 12/522,171.

[26] Pascal Fua. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In *Proceedings of the 12th international joint conference on Artificial intelligence*, pages 1292–1298, 1991.

[27] Timothy J Hall, Mehmet Bilgen, Michael F Insana, and Thomas A Krouskop. Phantom materials for elastography. *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on*, 44(6):1355–1365, 1997.

[28] EW Hart. Theory of the tensile test. *Acta Metallurgica*, 15(2):351–355, 1967.

[29] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. *Surface reconstruction from unorganized points*, volume 26. ACM, 1992.

[30] Thanarat Horprasert, David Harwood, and Larry S Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *IEEE ICCV*, volume 99, pages 1–19, 1999.

[31] Lenka Jerabkova. *Interactive cutting of finite elements based deformable objects in virtual environments*. PhD thesis, Universitätsbibliothek, 2007.

[32] Jörgen Kajberg and G Lindkvist. Characterisation of materials subjected to large strains by inverse modelling based on in-plane displacement fields. *International Journal of Solids and Structures*, 41(13):3439–3459, 2004.

[33] Rudolph Emil Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[34] M Kauer, V Vuskovic, J Dual, G Szekely, and M Bajka. Inverse finite element characterization of soft tissues. *Medical Image Analysis*, 6(3):275, 2002.

[35] K Konolige and P Mihelich. Technical description of kinect calibration, 2011.

[36] S KUBO. Inverse problems related to the mechanics and fracture of solids and structures. *JSME International journal. Series 1, Solid mechanics, strengh of materials*, 31(2):157–166, 1988.

[37] Jochen Lang. *Deformable model acquisition and validation*. PhD thesis, The University of British Columbia, 2001.

[38] Jochen Lang and Sheldon Andrews. Measurement-based modeling of contact forces and textures for haptic rendering. *Visualization and Computer Graphics, IEEE Transactions on*, 17(3):380–391, 2011.

[39] Jochen Lang and Dinesh K Pai. Estimation of elastic constants from 3d range-flow. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 331–338. IEEE, 2001.

[40] Jochen Lang, Dinesh K Pai, and Robert J Woodham. Acquisition of elastic models for interactive simulation. *The International Journal of Robotics Research*, 21(8):713–733, 2002.

[41] H Lee, Mark Foskey, Marc Niethammer, Pavel Krajcevski, and M Lin. Simulation-based joint estimation of body deformation and elasticity parameters for medical image analysis. 2012.

[42] Huai-Ping Lee and Ming C Lin. Fast optimization-based elasticity parameter estimation using reduced models. *The Visual Computer*, 28(6-8):553–562, 2012.

[43] Xiangyun Liao, Zhiyong Yuan, Zhaoliang Duan, Weixin Si, Si Chen, Sijiao Yu, and Jianhui Zhao. A robust physics-based 3d soft tissue parameters estimation method for warping dynamics simulation. In *AsiaSim 2012*, pages 205–212. Springer, 2012.

[44] Kevin Lister, Zhan Gao, and Jaydev P Desai. Development of in vivo constitutive models for liver: application to surgical simulation. *Annals of biomedical engineering*, 39(3):1060–1073, 2011.

[45] Bryn A Lloyd, Gábor Székely, and Matthias Harders. Identification of spring parameters for deformable object simulation. *Visualization and Computer Graphics, IEEE Transactions on*, 13(5):1081–1094, 2007.

[46] Rizwan Macknojia, Alberto Chavez-Aragon, Pierre Payeur, and Robert Laganiere. Experimental characterization of two generations of kinect's depth sensors. In *Robotic and Sensors Environments (ROSE), 2012 IEEE International Symposium on*, pages 150–155. IEEE, 2012.

[47] Lawrence E Malvern. *Introduction to the Mechanics of a Continuous Medium*. Number Monograph. 1969.

[48] Hatef Mehrabian and Abbas Samani. Constrained hyperelastic parameters reconstruction of pva (polyvinyl alcohol) phantom undergoing large deformation. In *SPIE Medical Imaging*, pages 72612G–72612G. International Society for Optics and Photonics, 2009.

[49] Marc Christian Metzger, Marc Gissler, Matthias Asal, and Matthias Teschner. Simultaneous cutting of coupled tetrahedral and triangulated meshes and its application in orbital reconstruction. *International journal of computer assisted radiology and surgery*, 4(5):409–416, 2009.

[50] Eder Miguel, Derek Bradley, Bernhard Thomaszewski, Bernd Bickel, Wojciech Matusik, Miguel A Otaduy, and Steve Marschner. Data-driven estimation of cloth simulation models. In *Computer Graphics Forum*, volume 31, pages 519–528. Wiley Online Library, 2012.

[51] Karol Miller and Kiyoyuki Chinzei. Constitutive modelling of brain tissue: experiment and theory. *Journal of biomechanics*, 30(11):1115–1121, 1997.

[52] Judi Mohne. Virtual reality for health care: a survey. *Virtual reality in neuro-psycho-physiology: cognitive, clinical and methodological issues in assessment and rehabilitation*, 44(3), 1997.

[53] Matthias Müller, Julie Dorsey, Leonard McMillan, Robert Jagnow, and Barbara Cutler. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 49–54. ACM, 2002.

[54] Don Murray and James J Little. Using real-time stereo vision for mobile robot navigation. *Autonomous Robots*, 8(2):161–171, 2000.

[55] Vishvjit S Nalwa. *A guided tour of computer vision*, volume 7. Addison-Wesley New York, 1993.

[56] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically based deformable models in computer graphics. In *Computer Graphics Forum*, volume 25, pages 809–836. Wiley Online Library, 2006.

[57] Han-Wen Nienhuys. Cutting in deformable objects. *Institute for Information and Computing Sciences, Utrecht University*, 2003.

[58] J Ophir, SK Alam, B Garra, F Kallel, E Konofagou, T Krouskop, and T Varghese. Elastography: ultrasonic estimation and imaging of the elastic properties of tissues. *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, 213(3):203–233, 1999.

[59] Dinesh K Pai, Jochen Lang, John Lloyd, and Robert J Woodham. Acme, a telerobotic active measurement facility. In *Experimental Robotics VI*, pages 391–400. Springer, 2000.

[60] Guillaume Picinbono, Herve Delingette, and Nicholas Ayache. Nonlinear and anisotropic elastic soft tissue models for medical simulation. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1370–1375. IEEE, 2001.

[61] James Duncan Revell. *Computer vision elastography.* PhD thesis, University of Bristol, 2005.

[62] Hassan Rivaz, Emad Boctor, Pezhman Foroughi, Richard Zellars, Gabor Fichtinger, and Gregory Hager. Ultrasound elastography: a dynamic programming approach. *Medical Imaging, IEEE Transactions on*, 27(10):1373–1377, 2008.

[63] Abbas Samani, Jonathan Bishop, and Donald B. Plewes. A constrained modulus reconstruction technique for breast cancer assessment. *Medical Imaging, IEEE Transactions on*, 20(9):877–885, 2001.

[64] Kiattisak Sangpradit, Hongbin Liu, Lakmal D Seneviratne, and Kaspar Althoefer. Tissue identification using inverse finite element analysis of rolling indentation. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 1250–1255. IEEE, 2009.

[65] DS Schnur and Nicholas Zabaras. An inverse method for determining elastic material properties and a material interface. *International Journal for Numerical Methods in Engineering*, 33(10):2039–2057, 1992.

[66] Jean-Marc Schwartz, Marc Denninger, Denis Rancourt, Christian Moisan, Denis Laurendeau, et al. Modelling liver tissue properties using a non-linear visco-elastic model for surgery simulation. *Medical Image Analysis*, 9(2):103, 2005.

[67] Jan Smisek, Michal Jancosek, and Tomas Pajdla. 3d with kinect. In *Consumer Depth Cameras for Computer Vision*, pages 3–25. Springer, 2013.

[68] G Soza, R Grosso, Ch Nimsky, P Hastreiter, R Fahlbusch, and G Greiner. Determination of the elasticity parameters of brain tissue with combined simulation and registration. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 1(3):87–95, 2005.

[69] Cedric Syllebranque, Samuel Boivin, Christian Duriez, and Christophe Chaillou. Estimation of hookean parameters of deformable bodies from real videos. In *Cyberworlds, 2007. CW'07. International Conference on*, pages 330–337. IEEE, 2007.

[70] Joseph Teran, Silvia Blemker, V Hing, and Ronald Fedkiw. Finite volume methods for the simulation of skeletal muscle. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 68–74. Eurographics Association, 2003.

[71] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *ACM Siggraph Computer Graphics*, volume 21, pages 205–214. ACM, 1987.

[72] Roger Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *Robotics and Automation, IEEE Journal of*, 3(4):323–344, 1987.

[73] Huamin Wang, James F O'Brien, and Ravi Ramamoorthi. Data-driven elastic models for cloth: modeling and measurement. *ACM Transactions on Graphics (TOG)*, 30(4):71, 2011.

[74] Zhongkui Wang and Shinichi Hirai. Modeling and property estimation of japanese sweets for their manufacturing simulation. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3536–3541. IEEE, 2010.

[75] Stefanie Wuhrer, Jochen Lang, Motahareh Tekieh, and Chang Shu. Finite element based tracking of deforming surfaces. *arXiv preprint arXiv:1306.4478*, 2013.

[76] Zhengyou Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1330–1334, 2000.

[77] Olgierd Cecil Zienkiewicz, Robert Leroy Taylor, and Jian Z Zhu. *The Finite Element Method: Its Basis and Fundamentals: Its Basis and Fundamentals*. Butterworth-Heinemann, 2005.