USING PHYSICS-BASED METHODS TO MODEL THE DEFORMATION OF ICE-HOCKEY STICKS FROM PLAYER SHOT VIDEOS

Leixiao Zhu

A thesis

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Computer Science

Concordia University

Montréal, Québec, Canada

December 2020

Concordia University

School of Graduate Studies

This is to certify that the thesis prepared

By: **Leixiao Zhu**

Entitled: **Using physics-based methods to Model the Deformation of Ice-Hockey Sticks from player shot videos**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

_____ Chair
Dr. Olga Ormandjieva

_____ Examiner
Dr. Olga Ormandjieva

_____ Examiner
Dr. Sudhir Mudur

_____ Co-supervisor
Dr. Abbas Javadtalab

_____ Co-supervisor
Dr. Tiberiu Popa

Approved _____
Dr. Leila Kosseim, Graduate Program Director

_____ 2020 _____
Dr. Mourad Debbabi, Dean of Faculty of Engineering and Computer Science

Abstract

Using physics-based methods to Model the Deformation of Ice-Hockey Sticks from player shot videos

Leixiao Zhu

In ice-hockey games, the hockey stick is usually deformed significantly when the players hit the puck. Different materials are used to make hockey sticks. Studies have been proposed to find how different hockey sticks and their deformation affect the performance of players. The reconstruction of a hockey stick's deformation during a shot is the key of many of those studies. The goal of this thesis is to model the deformation of the sticks from player shot videos using physics-based models and compare two physics-based methods for their performance on this problem. In this thesis, we propose two physics-based models to deform the stick and we integrate the models into a pipeline for stick reconstruction. The contribution of this thesis is twofold: 1) Integrate the physics-based models to the pipeline including data pre-processing, denoising and establishing constraints. 2) Adapting, implementing and comparing two physics based models. We evaluate the results by overlapping the deformed template with reconstructed point clouds and comparing our results with the data from a MOCAP system.

Acknowledgments

First of all, I am heartily thankful to my supervisors Dr. Tiberiu Popa and Dr. Abbas Javadtalab, for their patient guidance and inspiration throughout my thesis work. They are respectful and erudite scholars. Without their help, I could not have completed my thesis.

I also would like to thank my groupmate Gaurav Handa for his help and pleasant co-operation. I also appreciate all the professors and classmates at Concordia University from the classes I took for helping me expand my professional knowledge.

Lastly, I want to express my deep thanks to my parents for their encouragement and support all the time.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem and motivations

The Ice hockey game is a popular team sport, especially in Canada, the United States, and some European countries. During a shot, the hockey stick is usually significantly deformed. Since the hockey stick is the most important equipment in an ice hockey game, the stick and its deformation have been studied for decades. Studies such as[2][3] have been proposed to find how the material properties, such as the shaft stiffness, affect the speed and accuracy of a shot. In those studies, various methods are used to capture the deformation or the curvature of bending.

In many of these studies, an optical motion capture system with markers is applied to capture the deformation, e.g.[4]. Although those specific high-end devices come with high accuracy, they are usually costly and non-portable. Moreover, most of the systems only capture the curvature of bending since they only capture the trace of the markers. The 3D reconstruction of the hockey stick is important for stick performance analysis for its intuitiveness. However, not much work has been reported on this problem. Our team proposed a portable and low-cost framework to reconstruct the 3D shape of a hockey stick during a shot[5]. We employed a stereo camera set, which consists of two 275fps high-speed cameras, to capture the video of a shot. Then point clouds are reconstructed from the stereo videos for every frame, and the stick part is separated from the whole scene. A stick template is

aligned to the stick point cloud. The next step is to set constraints, and finally the template is deformed using physical based methods. Setting the constraints and deforming the template are essential parts in our framework since it directly affects the accuracy of the final result and the efficiency of the whole system. Therefore, the main goal of this thesis is to implement a robust physics-based module in our pipeline to set the constraints properly and deform the template to match the player videos.

Both setting constraints and deforming the template using physics-based models are challenging. For our portable stereo setup, in order to accomplish at least 250 fps, the images will be noisy and in low-resolution, which makes it hard to set constraints to the right place. Meanwhile, the hockey stick is a 3D object, but it is actually hollow and thin. It can be approximated as a 3D volumetric object or a thin shell object with no thickness. Therefore, for our physics-based model, we applied both 3D FEM and an energy minimizing method based on discrete shells for comparison.

1.2 Contribution

In this thesis, we integrate, adapt and test two physics-based deformation models to an end-to-end hockey acquisition pipeline that reconstructs the deformation of a hockey stick during a shot. We implement a module of this pipeline that sets the constraints, implements the physics-based models, and provides solvers to get the deformed shape. We test the system on 4 different image sequences with 2 different hockey stick templates and 2 different shot styles. The main contributions of this thesis are as follows:

- Integrate the physical models to the pipeline including data pre-processing, denoising, and establishing constraints.
- Adapting, implementing and comparing two physics based models.

1.3 Thesis Organization

The rest of the thesis is organized as follows:

- Chapter 2 : Background knowledge. The template-based reconstruction and multi-view stereo reconstruction are introduced. Then the physics knowledge of elastic material and linear elasticity is presented. The main approaches for simulating elastic materials and shell objects are introduced and discussed.

- Chapter 3 : Related works. The co-rotational Finite element method and an energy-based method named the discrete shells are presented. The least squares method used for both constraint setting and energy minimizing is also introduced.

- Chapter 4 : The methodology. The methodology is introduced in 4 parts: pre-processing of the data, building physics models, setting constraints, and solving the system for the physics models.

- Chapter 5 : Results and conclusions.

Chapter 2

Background knowledge

In this chapter, background knowledge is listed as follows. Firstly, the template-based reconstruction and multi-view stereo reconstruction are introduced. Then the background knowledge of elasticity, Hooke's law, and the relation between strain and stress are presented. Lastly, some commonly used methods for simulating elastic materials are introduced and discussed.

2.1 Template-based reconstruction

3D reconstruction from images and videos has been a hot topic for many years. A lot of methods have been proposed and many of them are template-based methods. In template-based reconstruction, a template of the non-deformed shape is provided. The input can be single images[6], RGB-D input[7], stereo image sequences, and so on. Then different methods such as motion priors and physics-based methods are applied to deform the template to match the real objects. Since the hockey stick is a typical elastic object which follows Hooke's law[8], we adopt and implement physics-based models to deform the template.

One of the main challenges of template-based reconstruction is to find a proper method to establish constraints between input data and the template. In our framework, we adopt a stereo camera setup for acquiring the player shot videos. Then we use multi-view stereo reconstruction to reconstruct the point cloud of the whole scene. The template is aligned to the point cloud of hockey stick and the constraint points are projected to the curve of

the center axis of the point cloud.

2.1.1 Multi-view stereo reconstruction

Multi-view stereo reconstruction is a technique that reconstructs the 3D structure of an object or the scene geometry from a series of images that are captured from stereo camera setups. In stereo reconstruction, the images are taken from different angles by different cameras at the same time. Then depth maps are computed for all the views and they are merged together to form the 3D model[9]. The multi-view stereo reconstruction can generate robust and accurate results. Besides, it is also used for generating input data or pre-processing in other works of 3D reconstruction, e.g.[10].

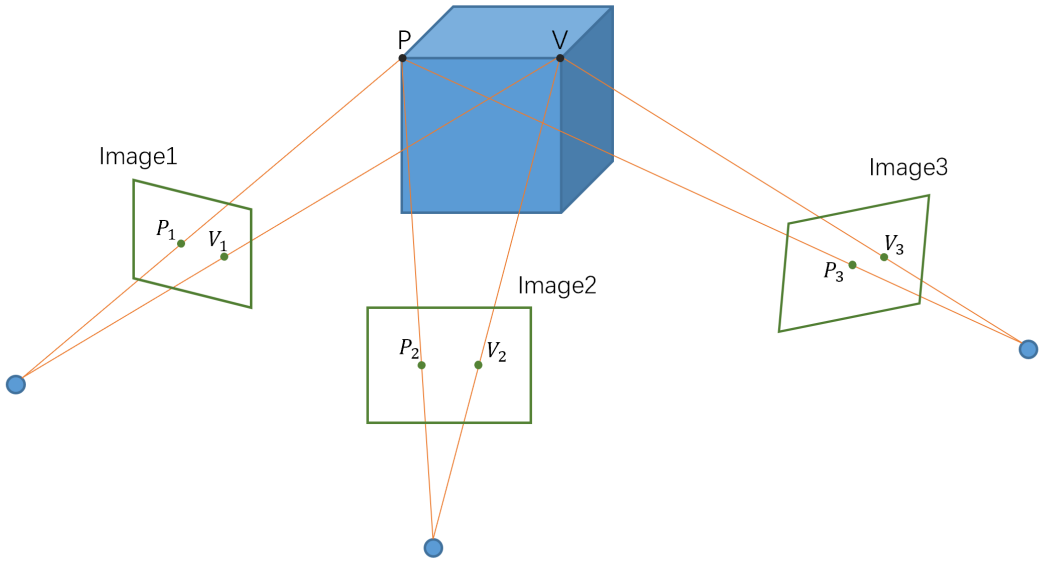

Figure 1: A Schematic of Multi-view Stereo System

2.2 Elasticity

When an object or a material is elastic, it will be deformed when applied external force and will return to its original shape when the force is removed[11]. When an elastic material is deformed, there are always internal forces against the deformation. The ability to resist

deformation is different for different materials. This ability determines the relationship between the internal force and deformation.

To describe this material property, the deformation is usually measured as strain (ϵ), and the force is measured as stress (σ). Strain describes the change of length or volume. It is defined as the differential of the deformed distance between the deformed shape and the original shape per unit length:

$$\varepsilon = \frac{\partial}{\partial X}(x - X) \tag{1}$$

where x is the deformed shape and X is the original shape.

Stress describes the force that resists the deformation. Generally it is defined as force per unit area:

$$\sigma = \frac{F}{A} \tag{2}$$

2.2.1 Linear Elasticity and Hooke's law

For most elastic materials, if the external force does not exceed the elastic limits, the relationship between force and deformation, or strain and stress, is linear.

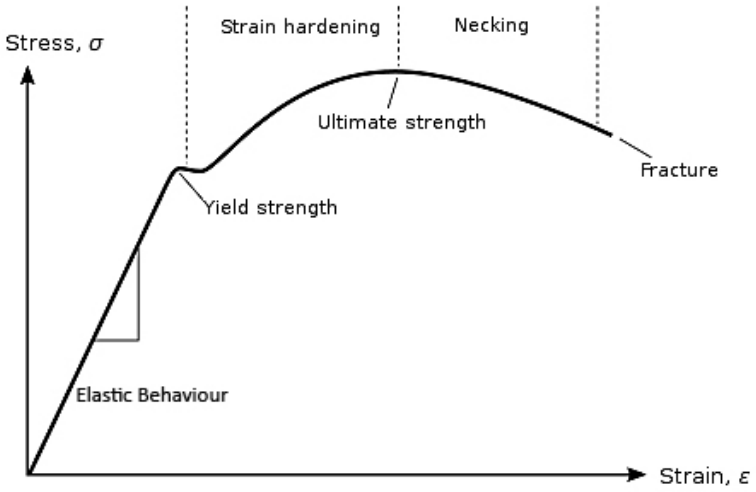

Figure 2: The strain-stress curve.[1]

This is known as Hooke's law, and those materials are called linear-elastic or Hookean materials. The deformation of a hockey stick when it hits the puck or the ground is a typical case of linear elasticity. According to Hooke's law, the relationship between internal force and the deformation can be expressed as $F = KX$, where F is the internal force that resists the deformation, X is the deformation, K is the stiffness parameter which only depends on the material property.

In 3 dimensions, the coefficient K becomes a stiffness matrix consists of 9 k values to take all directions into account. The formula of a spring in Hooke's law in three dimensions is:

$$F = \begin{bmatrix} K_{xx} & K_{xy} & K_{xz} \\ K_{yx} & K_{yy} & K_{yz} \\ K_{zx} & K_{zy} & K_{zz} \end{bmatrix} \begin{bmatrix} X_x \\ X_y \\ X_z \end{bmatrix} \tag{3}$$

$$F_x = K_{xx}X_x + K_{xy}X_y + K_{xz}X_z$$
$$F_y = K_{yx}X_x + K_{yy}X_y + K_{yz}X_z$$
$$F_z = K_{zx}X_x + K_{zy}X_y + K_{zz}X_z$$

2.2.2 Young's modulus and Poisson's ratio

The relationship between strain and stress is defined as a modulus. An elastic modulus is the slope of the stress-strain curve in the elastic deformation region[12]. Generally, three types of modulus are most commonly used for elastic deformation. They are Young's modulus, the shear modulus, and the bulk modulus. Young's modulus describes the resistance to the deformation along the axis where forces are applied. The shear modulus describes the resistance to shear. The bulk modulus describes the resistance to volume change. Besides those moduli, the Poisson's ratio is also introduced to describe the expansion or contraction on one axis when the material is compressed or stretched on the perpendicular axis. The four parameters, Young's modulus E, the shear modulus G, the bulk modulus K and the Poisson's ratio v are transformable with the equation:

$$E = 2G(1 + v) = 3K(1 - 2v) \tag{4}$$

Any two of the four parameters can be used to describe an isotropic material. An isotropic material means its resistance to deformation is the same in all directions. The most commonly used pair is the Young's Modulus and the Poisson's ratio.

2.2.3 Cauchy Strain tensor

In linear elasticity, the Cauchy strain[13] (also named engineering strain) is one of the most commonly used strain measurements. The definition of Cauchy strain is the deformed distance per unit length: $\varepsilon = \frac{u}{\mathrm{L}_0}$, where $u = L - \mathrm{L}_0$. In 3-D, the Cauchy strain tensor in matrix form is expressed as:

$$\varepsilon = \frac{du}{dx} = \begin{Bmatrix} \frac{\mathrm{d}u_x}{dx} & \frac{1}{2}(\frac{\mathrm{d}u_x}{dy} + \frac{\mathrm{d}u_y}{dx}) & \frac{1}{2}(\frac{\mathrm{d}u_x}{dz} + \frac{\mathrm{d}u_z}{dx}) \\ \frac{1}{2}(\frac{\mathrm{d}u_x}{dy} + \frac{\mathrm{d}u_y}{dx}) & \frac{\mathrm{d}u_y}{dy} & \frac{1}{2}(\frac{\mathrm{d}u_y}{dz} + \frac{\mathrm{d}u_z}{dy}) \\ \frac{1}{2}(\frac{\mathrm{d}u_x}{dz} + \frac{\mathrm{d}u_z}{dx}) & \frac{1}{2}(\frac{\mathrm{d}u_y}{dz} + \frac{\mathrm{d}u_z}{dy}) & \frac{\mathrm{d}u_z}{dz} \end{Bmatrix} = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{xz} \\ \varepsilon_{yx} & \varepsilon_{yy} & \varepsilon_{yz} \\ \varepsilon_{zx} & \varepsilon_{zy} & \varepsilon_{zz} \end{bmatrix} \tag{5}$$

The ε_{ij} where i = j means normal strain and where $i \neq j$ means shear strain. Note that $\varepsilon_{ij} = \varepsilon_{ji}$, so the final formula can be simplified as a 6-component vector:

$$\varepsilon = \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ 2\varepsilon_{xy} \\ 2\varepsilon_{yz} \\ 2\varepsilon_{xz} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = Bu \tag{6}$$

2.2.4 Cauchy stress tensor

Cauchy stress[13] is also a most commonly used stress in measuring linear elasticity. It is defined as $\sigma = E\varepsilon$, Where E is a tensor of elastic modulus. Same as the strain tensor, the stress tensor can be simplified to a 6-component vector. Therefore, E becomes a matrix of 6x6. For isotropic materials, using the Young modulus and the Poisson's ratio, the E

matrix is:

$$E = \frac{\lambda}{(1+v)(1-2v)} \begin{bmatrix} 1-v & v & v & 0 & 0 & 0 \\ v & 1-v & v & 0 & 0 & 0 \\ v & v & 1-v & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2}-v & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}-v & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}-v \end{bmatrix} \tag{7}$$

2.3 Commonly used methods for elastic material simulation

The linear elasticity we discussed above is in continuous form. However, in practice, the continuous problems need to be divided into finite pieces. Based on different ways of discretizing or representing the elastic material, different approaches have been proposed. The most commonly used methods are the Spring-Mass system, the Finite Element Method and energy minimizing methods.

2.3.1 Mass-Spring system

In reality, a spring is a typical Hookean model. In Mass-Spring system, the elastic material is represented as a set of points with mass connected by massless springs[14]. Then the forces are gathered for the points, e.g., force from spring, gravity, wind, collision, etc. For the system in 3D, the springs consist of 3 types: structural springs, shear springs, and flexion springs. The structural spring connects adjacent points and gives the tensile force. The shear spring connects points that are on a diagonal and gives the shear force. The flexion springs connect a point and the point after its adjacent point column-wise or row-wise, and gives the bending force. Figure 3 gives a picture of the three types of springs.

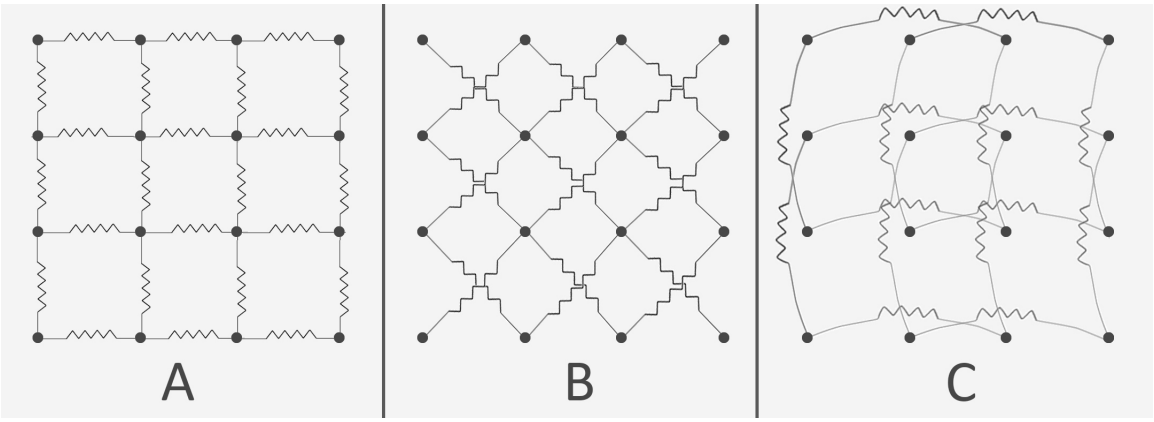

Figure 3: A: The structural springs. B: The shear springs. C: The flexion springs

The forces of springs are calculated using Hooke's law: $F = K\Delta L$, where ΔL is the length change of the springs. By applying Newton's second law $F = m\frac{dv}{dt} = ma$, a differential equation system can be obtained for the displacement of all the points. Then standard numerical schemes such as explicit or implicit integration can be used to solve the system. The Spring-Mass system is widely used for simulating objects with spring-like behaviour, such as hair[15] and clothes[16].

2.3.2 Finite Element Method

Among the methods for simulating elastic materials, the Finite Element Method is one of the most widely used method in computational physics and engineering[17]. Compared to the Mass-Spring system, it is more complex and more physically accurate, especially for modeling 3D volumetric problems.

The idea of Finite Element Method first appeared in early works in 1940s[18][19] and has developed over years[20]. The method subdivides a continuous geometry or a domain into finite parts. Those finite parts are called finite elements. Each of the elements is assigned with a local equation and those equations are then gathered into an equation system that models the entire problem[21].

Subdivision units

Since the Finite Element Method subdivides continuum problems into finite elements, the choice of element unit will affect the accuracy of the approximate solution. In FEM, continuum materials are usually divided into units like triangles, quadrangles, tetrahedrons, hexahedrons, and so on. Triangles and rectangles are the mostly used ones for deformation on surfaces, such as cloths and 2D objects. Triangle elements are more flexible, easier to implement, while quadrangle elements deliver more accurate results[22]. Tetrahedrons and hexahedrons are widely used for 3D problems. The tetrahedron elements are easier for subdivision while the hexahedron elements are more accurate in solving the system[23].

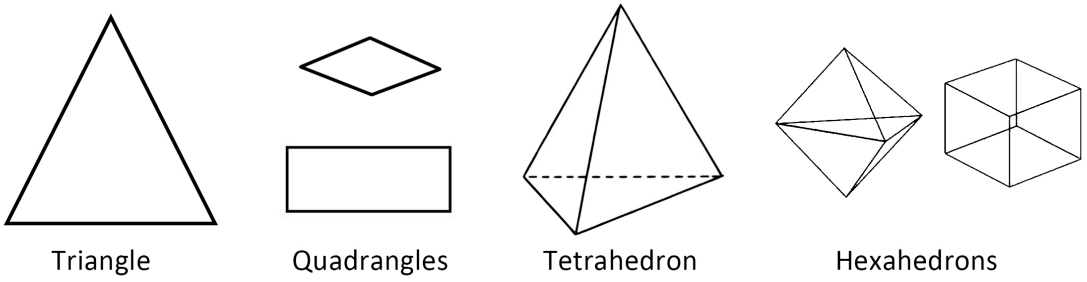

Figure 4: Different element units

Shape Function

To discretize the continuous strain-stress formula, the relation between global coordinates and element coordinates must be established. The function to transfer element local coordinates to global coordinates is called the shape function. Take 1D line segment element as example: suppose A and B are the positions of two endpoints of a line segment, then the position of any points on the line segment can be represented as:

$$C = \xi A + (1-\xi)B = \begin{bmatrix} \xi & 1-\xi \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix}, \tag{8}$$

Where $0 \leq \xi \leq 1$.

$(\xi, 1-\xi)$ is the local coordinate for C, and the function $P = \begin{bmatrix} \xi & 1-\xi \end{bmatrix}$, which represents the position of C with the positions of A and B is called shape function. Then the continuous form of Cauchy Strain can be discretized by the shape function:

$$\varepsilon_e = \frac{\mathrm{du}_e}{dx} = \frac{\mathrm{du}_e}{d\xi}\frac{d\xi}{dx} \tag{9}$$

Linear tetrahedron shape function

The tetrahedron, also known as the triangular pyramid, consists of four triangular faces, 4 vertices, and 6 edges. The linear tetrahedron shape function or the tetrahedral coordinate is a function defined to obtain the elastic properties of an arbitrary particle inside a tetrahedron by interpolating between the 4 vertices.

The tetrahedral coordinate or the tetrahedral shape function uses 4 scalars to defined a point inside a tetrahedron $(\xi_1,\ \xi_2,\ \xi_3,\ \xi_4)$. If the positions of 4 vertices are $v_1(v_{1x},\ v_{1y},\ v_{1z})$, $v_1(v_{2x},\ v_{2y},\ v_{2z})$, $v_3(v_{3x},\ v_{3y},\ v_{3z})$, $v_4(v_{4x},\ v_{4y},\ v_{4z})$, the position of V can be obtained by:

$$V = \xi_1 * v_1 + \xi_2 * v_2 + \xi_3 * v_3 + \xi_4 * v_4 \tag{10}$$

In matrix from:

$$\begin{bmatrix} 1 \\ x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ v_{1x} & v_{2x} & v_{3x} & v_{4x} \\ v_{1y} & v_{2y} & v_{3y} & v_{4y} \\ v_{1z} & v_{2z} & v_{3z} & v_{4z} \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \\ \xi_4 \end{bmatrix} \tag{11}$$

Where

- $\xi_1 + \xi_2 + \xi_3 + \xi_4 = 1$
- $0 \leq \xi_i \leq 1$

Otherwise, the vertex is outside of the tetrahedron.

$$P = \begin{bmatrix} 1 & 1 & 1 & 1 \\ v_{1x} & v_{2x} & v_{3x} & v_{4x} \\ v_{1y} & v_{2y} & v_{3y} & v_{4y} \\ v_{1z} & v_{2z} & v_{3z} & v_{4z} \end{bmatrix} \tag{12}$$

The function P (12) shows the partial derivatives of global coordinate respective to tetrahedron coordinate $\frac{dx}{d\xi}$. The inverse of function P, the function N, transforms Cartesian coordinate to tetrahedral coordinate.

$$N = P^{-1} = \frac{1}{6V}\begin{bmatrix} V_1 & a_1 & b_1 & c_1 \\ V_2 & a_2 & b_2 & c_2 \\ V_3 & a_3 & b_3 & c_3 \\ V_4 & a_4 & b_4 & c_4 \end{bmatrix} \tag{13}$$

Where V is the volume, $V = V_1 + V_2 + V_3 + V_4$

$$\begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \\ \xi_4 \end{bmatrix} = \frac{1}{6V}\begin{bmatrix} V_1 & a_1 & b_1 & c_1 \\ V_2 & a_2 & b_2 & c_2 \\ V_3 & a_3 & b_3 & c_3 \\ V_4 & a_4 & b_4 & c_4 \end{bmatrix}\begin{bmatrix} 1 \\ x \\ y \\ z \end{bmatrix} \tag{14}$$

The function N (13) shows the partial derivatives of tetrahedron coordinate respective to global coordinate $\frac{d\xi}{dx}$

Tetrahedral Strain

From equation (11) and function P, we can find the partial derivatives $\frac{du_x}{d\xi_i}$ are u_{xi}. From equation (14) and function N, we can find the partial derivatives $\frac{d\xi_i}{dx}$ are $\frac{a_i}{6V}$.

$$\frac{du_x}{d\xi_i} = u_{xi}, \quad \frac{du_y}{d\xi_i} = u_{yi}, \quad \frac{du_z}{d\xi_i} = u_{zi}$$
$$\frac{d\xi_i}{dx} = \frac{a_i}{6V}, \quad \frac{d\xi_i}{dy} = \frac{b_i}{6V}, \quad \frac{d\xi_i}{dz} = \frac{c_i}{6V}$$

So combining with equation (9), the strain can be written as:

$$\varepsilon_e = \frac{1}{6V}\begin{bmatrix} a_1 & 0 & 0 & a_2 & 0 & 0 & a_3 & 0 & 0 & a_4 & 0 & 0 \\ 0 & b_1 & 0 & 0 & b_2 & 0 & 0 & b_3 & 0 & 0 & b_4 & 0 \\ 0 & 0 & c_1 & 0 & 0 & c_2 & 0 & 0 & c_3 & 0 & 0 & c_4 \\ b_1 & a_1 & 0 & b_2 & a_2 & 0 & b_3 & a_3 & 0 & b_4 & a_4 & 0 \\ 0 & c_1 & b_1 & 0 & c_2 & b_2 & 0 & c_3 & b_3 & 0 & c_4 & b_4 \\ c_1 & 0 & a_1 & c_2 & 0 & a_2 & c_3 & 0 & a_3 & c_4 & 0 & a_4 \end{bmatrix}\begin{bmatrix} u_{x1} \\ u_{y1} \\ u_{z1} \\ u_{x2} \\ u_{y2} \\ u_{z2} \\ u_{x3} \\ u_{y3} \\ u_{z3} \\ u_{x4} \\ u_{y4} \\ u_{z4} \end{bmatrix} = Bu^e \tag{15}$$

The tetrahedron stiffness matrix

Gathering all the equations above, we can obtain the final energy equation as :

$$u_e^T F = \int_{V_e} {\varepsilon_e}^T \sigma_e dV_e \tag{16}$$

Taking $\varepsilon^T = u^T B^T$ and $\sigma = E\varepsilon = EBu$ into equation (16):

$$u_e{}^T F = u_e{}^T \int_{V_e} B_e{}^T E_e B_e u_e dV_e \tag{17}$$

The E and B are matrices of constants. And for linear isotropic materials, since its elastic property is same in all directions, the $\int_{V_e} dV_e$ can be integrated into V. So the final equation can be written as:

$$F = V_e B_e{}^T E_e B_e u_e \tag{18}$$

The final equation establishes a linear relationship between force and deformation, which follows the Hooke's law $F = Kx$.

$$K_e = V_e B_e{}^T E_e B_e, \quad F = K_e u_e \tag{19}$$

2.3.3 Energy Based Methods

In Classical physics, the energy change of a system equals to the work made by external force. Work can be expressed as the product of the value of force and the displacement: $E = FU$. Similar to force, energy can also describe the status of elastic materials.
Take 1D spring as an example:

$$E = FU = \int_L^0 -KLdL = \frac{1}{2}KL^2 \tag{20}$$

Where L is the length change.

Energy-based methods are usually used for more complex situations and for more physically accurate models. For example, when a thin object experience curved deformation, the relationship between force and displacement is no longer linear and is hard to represent. The Spring-Mass system uses a linear flexion spring to approximately represent bending force. However, it is not physically accurate.
According to T. J. Willmore's bending energy functional[24], the bending energy of a surface can be expressed as:

$$E = \int_S (H^2 - K)dA, \ or \ E = \int_S H^2 dA \tag{21}$$

Where S is the surface domain, A is the area , $H = (k_1 + k_2)/2$ is the mean curvature, $K = k_1 k_2$ is the Gaussian curvatureand k_1, k_2 are the two principal curvatures of the surface.

2.4 Time integration and Energy minimizing in physics simulation

In physics simulation, time integration is a popular method for solving problems where people want to see intermediate animations. There are mainly two types of time integration methods: the Explicit (Forward) Euler Method and the Implicit (Backward) Euler Method[25]. The explicit Euler method uses the slope from the previous time step to estimate the result of the next time step:

$$y_{n+1} = y_n + \Delta t f(y_n, t_n) \tag{22}$$

The explicit Euler method is easy to implement. However, in order to reduce the errors, the time step should be as small as possible. Otherwise, the system may explode soon.
The implicit Euler method uses the slope from the next time step to estimate the result of the next time step:

$$y_{n+1} = y_n + \Delta t f(y_{n+1}, t_{n+1}) \tag{23}$$

The implicit Euler method can not solve the system directly, since it needs the estimation of next time step. The explicit Euler method could be used to estimate the result of the next time step.

For problems that do not need to show the intermediate animations and require higher accuracy, energy minimizing methods are usually used to solve the system, e.g.[26]. The idea of energy minimizing method is to find the condition that minimizes the energy change of the whole system. For elastic material simulation, the energy minimizing method is to find the shape that minimizes the energy of deformation. Based on different models, different regression methods can be used to solve the system. For example, for force-based linear

elastic models, such as linear finite method, to minimized energy is actually to minimize the force. In this case, linear least squares method can be used to solve the system. For energy based models, non-linear least squares method can be applied to solve the system.

Chapter 3

Related Works

In this chapter, two physics-based methods, the co-rotational Finite element method and the discrete shells method, are presented. The least squares method used both for setting constraints and solving physics models is also introduced.

3.1 co-rotational FEM

In 3D elastic material simulation, the finite element method with linear tetrahedron elements is widely used because of its simpleness for mesh subdivision and flexibility to fit different situations. However, since it is a linear approximation, the model is not rotationally invariant. The error will be large if the system experiences large rotations. The problem can be solved by using a non-linear strain-stress measurement such as the Neo-Hookean model[27][28]. However, non-linear methods are more complex and numerically unstable compared to linear measures. A method called stiffness warping technique is proposed to solve this problem for linear FEM. The method is proposed by Müller et. al[29]. The idea is to use a non-rotated reference frame for deformation, and then rotate the deformed reference frame to get the final result.

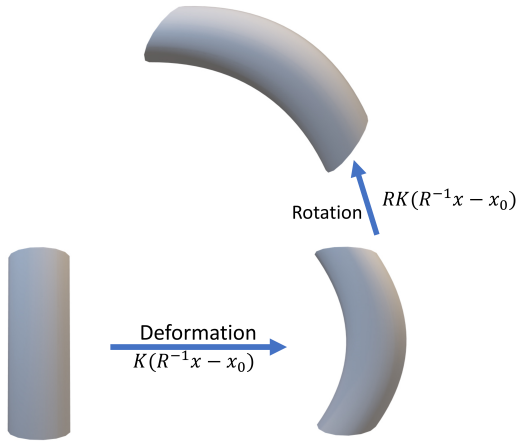

Figure 5: The process of co-rotational stiffness matrix

After applying the stiffness warping technique, The full equation for co-rotated FEM will be:

$$F = V_e R B_e{}^T E_e B_e (R^{-1}x - x_0) \tag{24}$$

The rotation matrix can be extracted from the shape functions of undeformed and deformed shapes[30]. Since the tetrahedron coordinate does not change with deformation (recall the shape function in chapter 2, the tetrahedron coordinate for a point inside a tetrahedron will always be the same), a linear relationship can be established between the deformed and undeformed state:

$$p = P\beta, \quad q = Q\beta \quad \Rightarrow \quad q = QP^{-1}p \tag{25}$$

Where p is the initial shape and q is the deformed shape. The matrix $A = QP^{-1}$ is an affine transform matrix and thus can be separated into rotation R, scaling S, and transform T:

$$A = QP^{-1} = \begin{bmatrix} RS & T \\ 0 & 1 \end{bmatrix} \tag{26}$$

Where RS is the combination of rotation and scaling, which is a 3x3 matrix and T is a 3x1 vector.

To extract the rotation matrix, a polar decomposition can be applied to transform matrix

T[31]:

$$A = USV^T, \quad R = VU^T \tag{27}$$

The R is a 3x3 matrix. Since it is an affine transformation, the rotation matrices for the 4 vertices of a tetrahedron are the same. Therefore, the rotation matrix for co-rotational stiffness matrix is:

$$R_k = \begin{bmatrix} R & 0 & 0 & 0 \\ 0 & R & 0 & 0 \\ 0 & 0 & R & 0 \\ 0 & 0 & 0 & R \end{bmatrix} \tag{28}$$

3.2 The Discrete Shell Energy

When calculating the elastic energy of a 3D thin shell object, a main challenge is to calculate the bending energy with a proper discretization method. Related works have been proposed such as Discrete Quadratic Curvature Energies[32] and *Discrete shells*[33]. We adopted the discretization method and energy functions from *Discrete shells*[33] for their intuitiveness. In their work, they proposed a discrete form of energies based on the triangles of a triangular mesh.

In their physics model for shell objects, the system consists of the energy of nonlinear membrane and bending energies. In physics, the membrane is a very thin layer of material. Here the nonlinear membrane means a surface with no thickness. The elastic membrane surface has resistance to stretching and shearing. Stretching is the local change in area, and Shearing is the local change in length while the area remains the same. In their work, the membrane energy of Stretching and shearing is expressed on the area and length of edges of the triangles:
The stretching energy:

$$E_{stretch} = \int_0^S K_{stretch} A^2 dA \approx \sum_S K_{stretch} (1 - \frac{A}{A_o})^2 A_o \tag{29}$$

Where A is the area of every triangle, A_o is the area of undeformed shape and S means area change of all the triangles.

The shearing energy:

$$E_{shear} = \int_0^L K_{shear} e^2 de \approx \sum_L K_{shear} (1 - \frac{e}{e_o})^2 e_o \tag{30}$$

Where e is the length of every edge, e_o is the undeformed length and L means the length change of all the edges.

For the discretization of bending energy, they applied shape operators which is the derivative of the Gauss map over the surfaces to evaluate the change in curvature. Then the change in curvature can be expressed as:

$$[Tr(\varphi * S) - Tr(S_o)]^2 = 4(H \circ \varphi - H_o)^2 \tag{31}$$

Where S and S_o are the shape operators of deformed and undeformed shape, H and H_o are the mean curvatures and φ is the affine transform function between the deformed state and undeformed state. Then integrate this equation over the shell mesh, the bending energy can be expressed as:

$$E_{bend} = \int_\Omega K_{bend} 4 \left(H \circ \varphi - H_o\right)^2 dA \approx \sum_e K_{bend} (\theta - \theta_o)^2 \frac{e_o}{h_o} \tag{32}$$

Where θ and θ_o are the angles of deformed shape and initial shape between two triangles over edge e, e_o is the length of edge e of undeformed shape and h_o is a third of the average heights of the two triangles over edge e.

Gathering the energy equations, the total elastic energy is:

$$E = \sum_S K_{stretch} (1 - \frac{A}{A_o})^2 A_o + \sum_L K_{shear} (1 - \frac{e}{e_o})^2 e_o + \sum_e K_{bend} (\theta - \theta_o)^2 \frac{e_o}{h_o} \tag{33}$$

3.3 Least Squares Method

The least squares method[34] is a commonly used approach for regression. One of the main usages of least squares is data fitting. It finds the approximate solution by minimizing the

sum of the squares of residuals of a system of equations. In our system, this method is applied to extract the curve of the center axis of the hockey stick from the captured data, as well as to deform the template by minimizing the total force or energy to match the captured data.

3.3.1 Linear least squares

A regression model is linear when the model is a linear combination of the parameters. For liner least squares, ordinary least squares (OLS) is the most common estimator:

$$\min_{x_0 \ldots x_m} \left\| \begin{pmatrix} k_{00} & \cdots & k_{0m} \\ \vdots & \ddots & \vdots \\ k_{n0} & \cdots & k_{nm} \end{pmatrix} \begin{pmatrix} x_0 \\ \vdots \\ x_n \end{pmatrix} - \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix} \right\|_2 = \min \|Ax - b\|_2 (34)$$

The mathematical solution of the linear least squares problem using OLS is:
If we have

$$f(x) = \sum_{j=1}^{m} K_j G_j(x) \tag{35}$$

Where the function $G_j(x)$ is a funciton of x. For expamle, $G_j(x) = x^j$. If $X_{ij} = G_j(x_i)$ and $y_i = f(x_i)$ are know and we need to find K_j, then the coefficients K_j can be obtained by[35]:

$$K = (X^T X)^{-1} X^T y \tag{36}$$

3.3.2 Non-linear least squares

When a system is not a linear combination of parameters, such as the bending energy, it can no longer be solved by linear least squares method. The general strategy for solving non-linear optimization problems is to solve a sequence of approximations to the original problem[36]. For non-linear least squares, we can split the equation using Taylor Expansion: $F(x + \Delta \mathrm{x}) \approx F(x) + J(x)\Delta \mathrm{x}$. Then the minimization problem can be approximated to:

$$\min_{\Delta\ x} \ \frac{1}{2} \|J(x)\Delta\ x + F(x)\|_2 \tag{37}$$

It is now a linear least squares problem for $\Delta\ x$. However, we cannot solve this system directly since the system may not converge properly with large step $\Delta\ x$. To have an accurate converge approximation, the delta X should be controlled under a certain size and then the

minimization procedure is iterated until the residual under a certain threshold.

There are mainly two approaches for controlling the size of $\Delta\, x$, the Trust Region method[37], and the Line Search method[38]. The trust region approach uses a simpler approximate function, usually a quadratic one, instead of the original function to find a local solution on a subset. The subset is named as the trust region. If the local solution successfully minimizes the original function, the trust region is then expanded. Otherwise, the trust region is contracted and solve the approximate function again. The line search approach first finds a descent direction, and then the step size is computed in that direction. Generally, the trust region method is more stable than the line search method.

Chapter 4

The Methodology

In this chapter, the proposed approaches for setting constraints and building the physics-based models are explained.

4.1 Overview

The proposed system consists of mainly 4 parts:

- The first part is pre-processing. This part consists of the pre-processing for captured data and pre-processing for the template. For the captured data, our team used two high-speed cameras to capture stereo image sequences. Then the point cloud of the whole scene is reconstructed, and the point set of the hockey stick is extracted from the point cloud[5]. For the template, a triangular mesh is generated from the original quadrilateral mesh. A tetrahedron file is also generated with the vertices of the triangular mesh.

- The second part is building and initializing the physics models. We used two different methods, the finite element method and the discrete shell method. The two methods have their own pros and cons. We used Graphite[39] as GUI and the API to load the template mesh. After loading the mesh, the initial positions of every vertex are stored. For the FEM, additional information of the tetrahedron is also stored and then the K matrices are built. For the discrete shell method, the vector of two points of every edge and the indices of the 4 vertices of every two adjacent triangles are stored to

build the energy equations.

- The Third part is setting constraints. The idea is to project constraint points from the template to the reconstructed point cloud. For denoising, the center axis of the point cloud is extracted. The center axis is split into the shaft part and blade part. Two quadratic polynomial curves are fitted to the two parts. Then we align the deformed template from the previous frame to the point cloud. The constraint points are then projected to the two curves.

- Finally, after the physics models are initialized and the constraints are set, the energy or the force is minimized using least squares solvers. We used a linear least squares solver for the linear FEM method and a non-linear least squares solver for the discrete shells method.

4.2 Pre-processing for the template

The original mesh of the hockey stick is an industrial model, which consists of hundreds of thousands of vertices and quadrilaterals. For the finite element method in computer graphics, more subdivided elements do not necessarily mean higher accuracy, restricted by the precision of floating-point numbers. However, more subdivided elements do mean more burden on calculation. Therefore, a step to reduce the number of vertices is essential before tetrahedralizing the mesh. There are several mature algorithms for re-meshing and mesh generating. We used Tetgen for generating the tetrahedron mesh as well as reducing the number of vertices. Tetgen is a mesh generator designed to subdivided any 3D mesh into tetrahedron elements[40]. After re-meshing, the final files are a triangular mesh with around 7000 vertices for both two physics models and a tetrahedron file for FEM.

4.3 Pre-processing for data

The reconstruction and extraction work is done by my teammate. After his work, the center axes of the hockey stick point cloud is extracted and fed into the physics engine. Although

most of the noise has been removed at the extraction step, there is still a chance that some outliers remain on the center axis. Therefore, it is necessary to remove all the outliers before fitting curves to the center axis. We used the Random sample consensus (RANSAC) to remove the outliers.

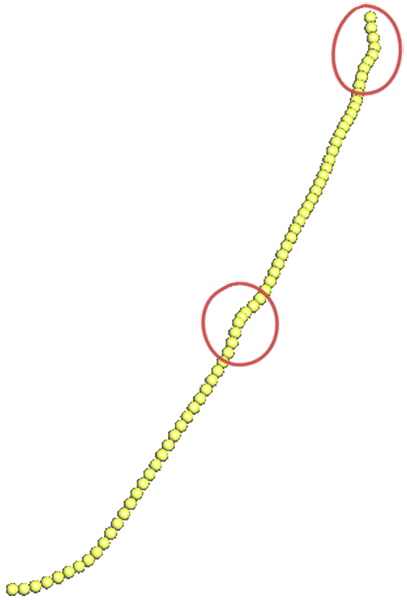

Figure 6: The outliers on the center axis

4.3.1 Random Sample Consensus method

Random sample consensus (RANSAC) is an iterative method to fit a mathematical model, such as a polynomial curve, to the data that contains the outliers that do not have a significant impact on the estimates[41]. In our case, the parameters of mathematical model are the coefficients of quadratic equations since we want to fit quadratic curves to the center axes. Therefore, the RANSAC algorithm in our case can be summarized as:

Repeat N iterations:

- Randomly select R points from the whole center axis. (R ¿ the degree of freedom of the curve)

- Fit a curve to the R points using linear least squares method.
- Given a threshold S, record the points which have a smaller distance to the curve than threshold S as inliers.

Compare the number of inlier points of every randomly selected point set. Find the point set which has the most inliers and output the inliers and outliers.

In our case, we repeat 30000 iterations, select 6 points, and set the threshold as 0.003. After the iterations are done, we get a clean center axis without outliers. Then a final curve is fitted to the inliers.

4.3.2 Polynomial curve fitting in 3D space

In the process of RANSAC, curve fitting is repeated every iteration. After RANSAC, a final curve is obtained by curve fitting on all the inliers. As introduced in chapter 2, the linear least squares method is a good choice for our system.

For curve fitting in 2D, it is easy to get the coefficients. Let the polynomial curve be $y = a_0 + a_1x^1 + a_2x^2 + \cdots + a_nx^n$, where the $a_0 \cdots a_n$ are the unknown coefficients. Suppose we have m number of known y and x:

$$b = \begin{bmatrix} y_0 \\ \vdots \\ y_n \end{bmatrix}, \quad A = \begin{bmatrix} \begin{pmatrix} 1 & \cdots & {x_1}^2 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & {x_n}^2 \end{pmatrix} \end{bmatrix}, \quad x = \begin{bmatrix} a_0 \\ \vdots \\ a_n \end{bmatrix}$$

In order to get the coefficients, we need to solve $\min \|Ax - b\|_2$. By using the Ordinary least squares equation (36), the coefficients can be obtained by $x = (A^TA)^{-1}A^Ty$.

However, for curve fitting in 3D, we can not simply choose x as the variable and find the coefficients for $Y = KX$ and $Z = KX$, because for a polynomial curve in 3D space, the relation between x, y and z may not be able to be represented by polynomial equations. Therefore, another variable t is introduced to the polynomial equations. The variable t

represents the order of the data. Therefore it can be time, the index, or a specific order. The equation for a 3D curve is then separated to x, y, and z dimension respectively:

$$F(t) = \begin{cases} X = a_{x0} + a_{x1}t + \ldots + a_{xn}t^n \\ Y = a_{Y0} + a_{Y1}t + \ldots + a_{Yn}t^n \\ Z = a_{Z0} + a_{Z1}t + \ldots + a_{Zn}t^n \end{cases} \tag{38}$$

In our system, because we want to fit a quadratic curve to the center axis, n should be 2.

4.3.3 Separate shaft part and blade part

After running RANSAC on the center axis, the shaft part of the center axis will be extracted, leaving outliers as well as the blade part. That is because the blade part is not on the same curve as the shaft part and is treated as outliers if we fit a quadratic curve to the center axis.

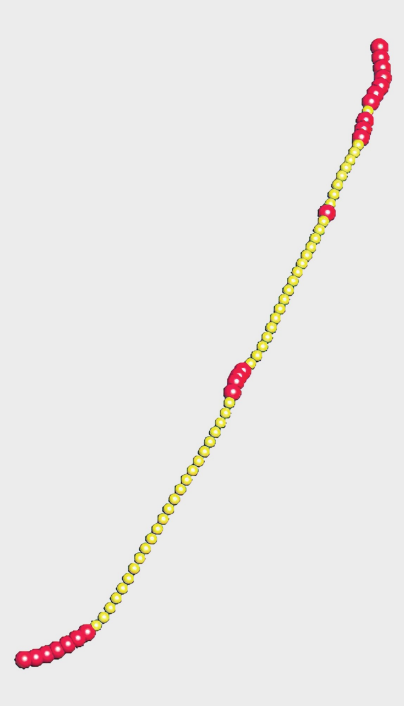

Figure 7: The blade part is also treated as outliers

The next step is to get the inliers of the blade part. With the shaft part extracted, we get the blade part with outliers. However, if we run RANSAC directly on the point set,

the outliers won't be removed, because the blade part is short, and the number of outliers from the shaft part may even exceed the number of blade points. As a result, the outliers start to affect the values of the estimates. A simple and efficient way to remove the outliers from the shaft part is to find the joint point between the blade and shaft and remove all the points above the joint point. Here the "above" means the direction from the joint to the top of the shaft, and the joint point is at the bottom of the shaft. Since the global coordinate is not calibrated after reconstruction, the coordinates of the two cameras and the average position of center axis are used to find the local coordinate. Let the coordinates of the camera on the left be C1, the camera on the right be C2, the average position be A, then the local positive y direction can be obtained by the cross product of vector $C2 - A$ and vector $A - C1$. With the calibrated coordinate, the joint point can be found by comparing their y coordinates.

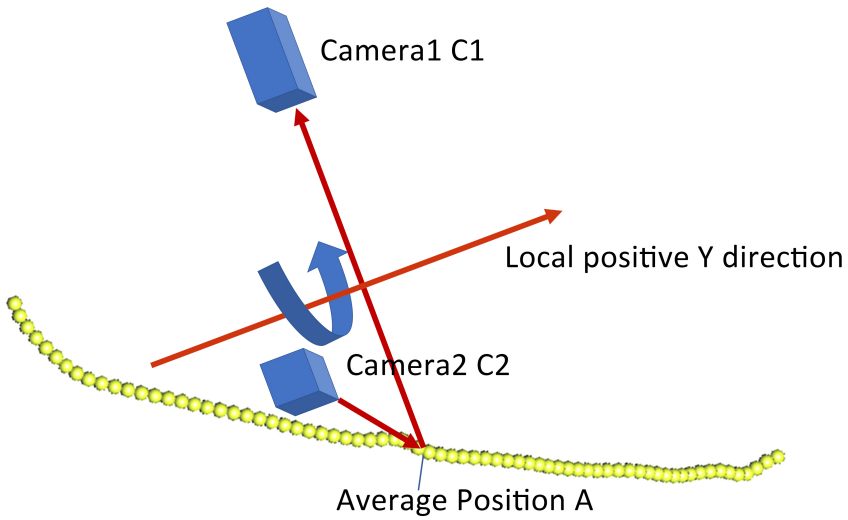

Figure 8: By the coordinate of two cameras and the center of point cloud, the calibrated Y direction can be found

4.4 Initializing the physics models

As mentioned in previous chapters, We adopted two models to implement the physics engine. First, we used the Linear Finite Element Method to simulation the hockey stick as a 3D volumetric object. Then we adopted an energy minimizing method with the energy functions from the Discrete Shells Method to simulate the hockey stick as a thin shell object.

4.4.1 Initialize system for Finite Element Method

The key for initializing the Finite Element Method is to calculate the stiffness matrix for every tetrahedron element. The initialization of the FEM can be summed as following steps:

- Load the template and the tetrahedron file.
- Record the initial position of every vertex and calculate the shape function for every tetrahedron.
- Calculate the Elastic modulus matrix E and the stiffness matrix $K = VB^TEB$ for every tetrahedron

Since we adopted the co-rotational stiffness method, the initial stiffness matrix K also needs to be recorded and will not change for later calculation. Then the co-rotational stiffness matrix $K_{corotate}$ in later iterations is obtained by RKR^{-1}.

4.4.2 Initialization of the thin shell energy minimizing method

The initialization of thin shell energy minimizing method is more complex. To calculate the bending energy $W_B(x) = k_B \sum_e (\theta_e - \theta_{e_o})^2 e_o/h_{e_o}$, the angles between every 2 neighboring triangles, the lengths of shared edge, and the heights relative to the sharing edge need to be calculated and stored for initial shape. We used a structure to store the information:

```
struct trianglePair{
    vector4int indices;
    float initialAngle;
    float initialEdgeLength;
```

float initialAverageHeight

}

Where the "indices" is an array of integers, which stores the indices of the 4 vertices of a triangle pair.

For shearing energy $W_L = \sum_e (1 - \frac{e}{e_o})^2 e_o$, the indices of the two vertices of each edge and the length are stored in a similar structure. For stretching energy $W_A = \sum_A (1 - \frac{A}{A_o})^2 A_o$, the indices and the area of every triangle are stored in the same way.

The length and the area are easy to calculate. As for the angle between two triangles, it is more complex. For arbitrary two vectors in 3D space, the angle between them ranges from 0-180 degrees. If people use the dot product to calculate the angle between the normal vectors of the two triangles, they will get an angle of that range. However, for arbitrary two neighboring triangles from a 3D mesh, the angle should range from 0-360 degrees, since there is another factor: "inside" or "outside".

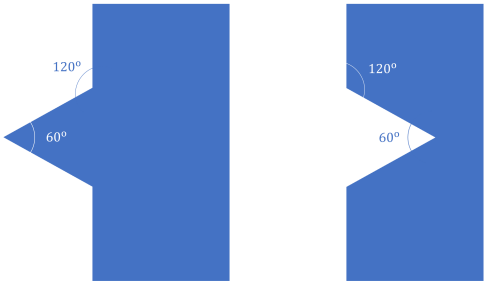

Figure 9: Although the angles between normal vectors are the same, there is another factor "inside" or "outside" for a 3D mesh

The figure 9 reveals the difference between "inside" and "outside". If we ignore this difference, the system may not converge properly. Here is an example showing a not correctly converged result:

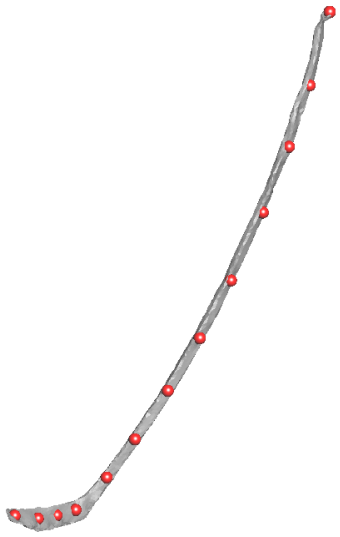

Figure 10: The red points are constraint points. The system converges globally,deforming the template by constraints; but does not converge locally, resulting in the uneven surface.

Calculate angles between 0-360 degree

To get a 0-360 degree angle, a third vector is needed to determine the clockwise direction. The third vector should be vertical to both the two vectors. With this vector, we can determine the clockwise or counterclockwise direction and an angle larger than 180 degrees can be obtained. For example, the angle in figure 11 is 240 degree clockwise.

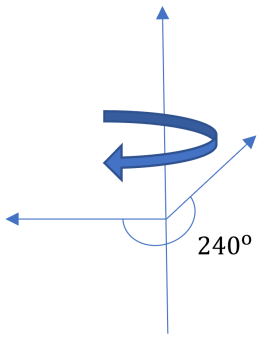

Figure 11: A third vector is used to determine the clockwise angle.

For a triangle pair $v1, v2, v3, v4$, since a normal is vertical to its surface, the edge $v2v3$ is vertical to both two normal vectors. Therefore, the angle is obtained by following steps:

1. Calculate the cross product of the two normal vectors of the two triangles as c.
2. Calculate the dot product of c and the vector of $(v2 - v3)$ as d.
3. If $d = -1$, that means the angle is over 180_o counterclockwise and the angle A is $A(x, y, z) = 2 * \pi - arccos(f(x, y, z))$, where $f(X, y, z)$ is the function to get the angle by dot product:$f(x, y, z) = \frac{v_1 \bullet v_2}{|v_1||v_2|}$
4. if $d = 1$, that means the angle is 180_o counterclockwise and the angle A is $A(x, y, z) = arccos(f(x, y, z))$.
5. if $d = 0$, that means the angle is 180_o and $A(x, y, z) = 180$.

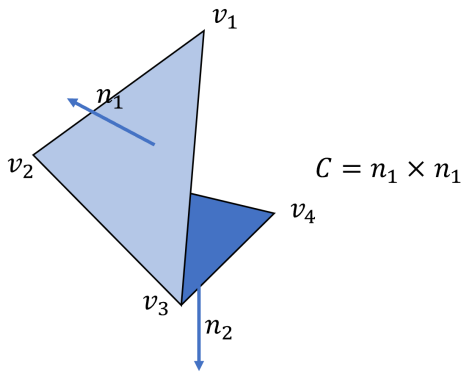

Figure 12: A triangle pair

4.5 Setting Constraints

The key idea is to project the points from the center axis of the template to the center axis of point cloud. However, there is no point inside the stick template, so we chose the center axis on one side. As a result, there will be an offset, but the deformed shape will be the same. The constraint points are currently manually selected. The rules for selecting constraint points are:

- Select all the points on the same side
- The constraint points should be as in the middle as possible.
- For the shaft part, the points should be approximately on the same straight line, since the original shape of the shaft is straight. For the blade part, the points should be approximately on a curve, since the original shape is curved.

The figure 13 shows how we selected the constraint points. Note that the points for the shaft part are not exactly in the middle because for an automatically generated triangular mesh, it is difficult to find a set of points that are in the middle and also on the same straight line at the same time. The priority for constraint points is that they should be approximately on the same line.

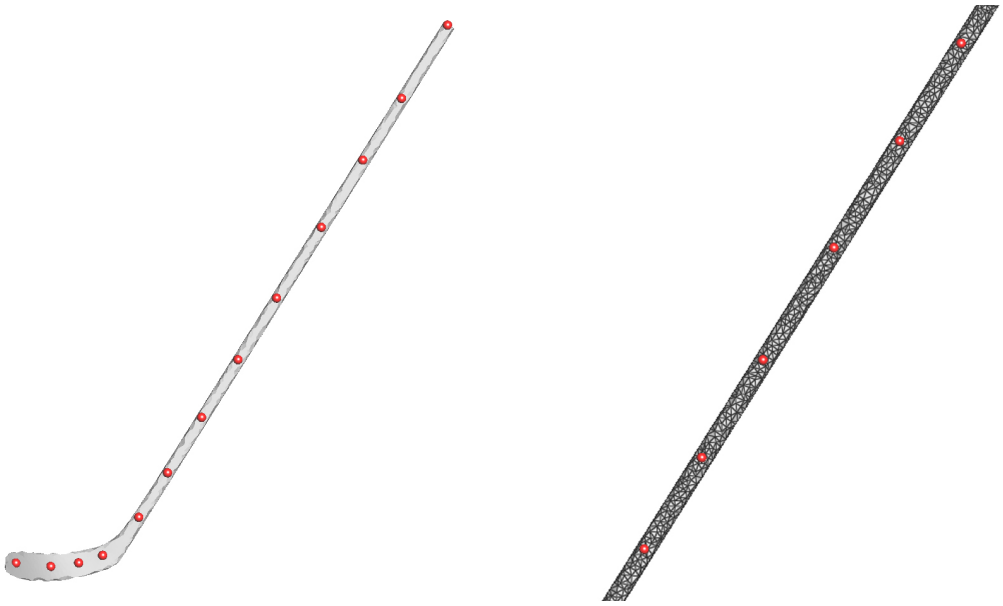

Figure 13: An example for selecting constraint points.

4.5.1 Adaptive alignment of template

After the indices of constraint points are selected, the template is aligned to the point cloud, and the constraint points are projected to the point cloud. The alignment of the template affects the accuracy of constraints when projected to the curve of the center axis. We used the Iterative closest point (ICP) method to align the template to the point cloud. The Iterative Closest Point method minimizes the distance between two point clouds[42].

For the first several frames where the stick is deformed slightly, the alignment is accurate. However, the error increases when the stick starts to bend intensely. We adopted an adaptive alignment step to solve this problem. That is to use the deformed template from the previous frame for alignment instead of the undeformed template. Since the bending curvature does not differ much between adjacent frames, the error will not increase a lot with the most bent frames. After the constraint points are projected to the targets, their positions are transferred to the undeformed template.

4.5.2 Projection Method

At first, our method is to project the constraint points to a curved surface which is fitted to the point set. We used the least squares method to fit a surface to the point set of the hockey stick and then project the points to the surface. The result was not satisfying because of the massive noise. The figure 14 shows the fitted surface, the constraints set to the surface, and the deformed template with the fitted surface.

Then we changed our method from surface fitting to extracting the center axis and curve fitting. It is far easier to remove noise on the center axis. By using RANSAC introduced before, most of the noise will be totally removed. The new method returns a promising result: Figure 15 shows the fitted curve and center axis, the constraints set to the curve, and the deformed template with the point cloud.

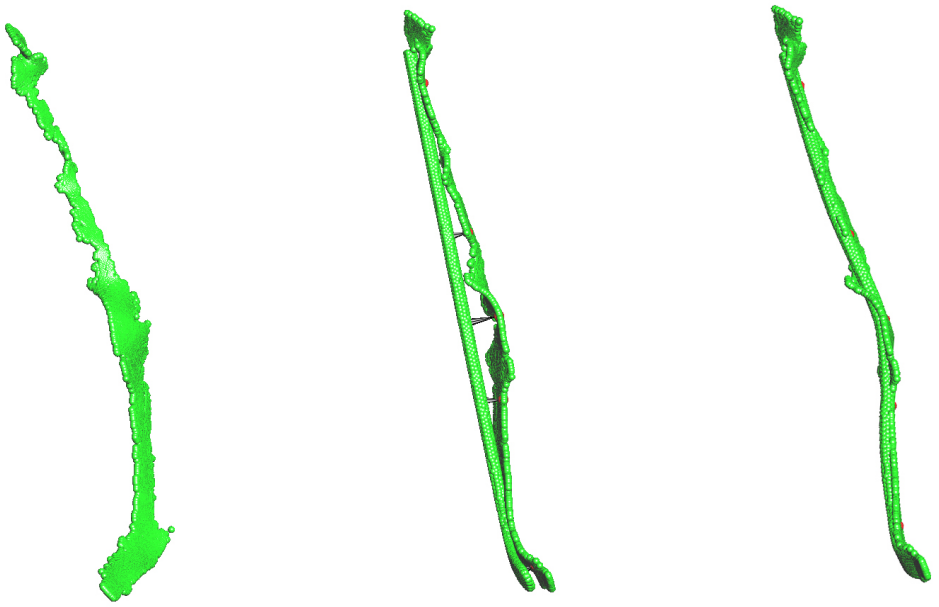

Figure 14: The result using surface fitting

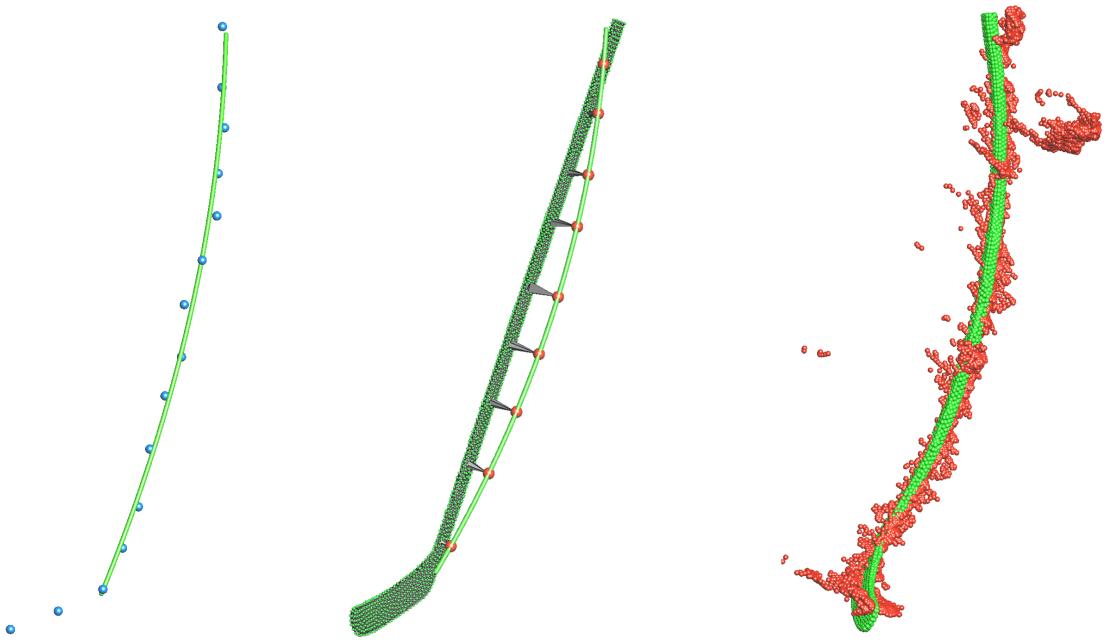

Figure 15: The result using center axis and curve fitting

Figure 16 is a comparison of the final results of using surface fitting and curve fitting:

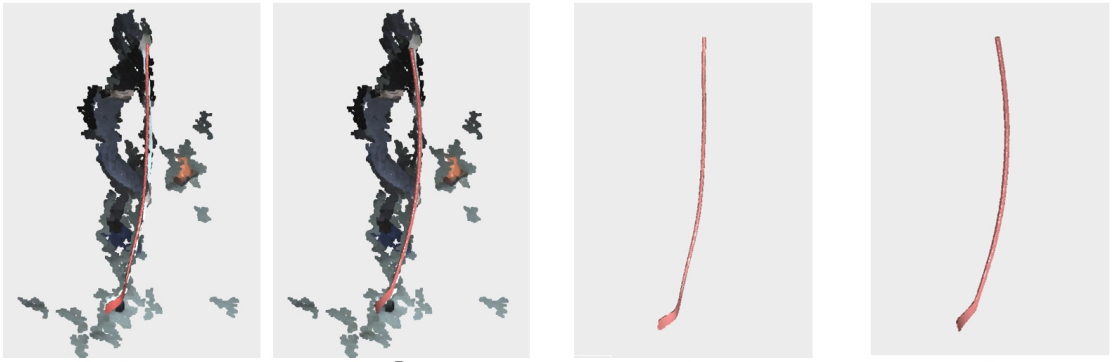

Figure 16: Comparison of using surface fitting and curve fitting.A is the result of using surface fitting. B is the result of extracting center axis and using curve fitting.

4.5.3 Project Constraint points

Projecting the constraint points means to find points on the curve which have the shortest distance to them. There are several ways to find the shortest distance and the corresponding point on the curve between a point and a curve, e.g., by regression and by derivative. Here we took the derivative way.

If we have a point (x, y, z) and a quadratic curve $F(t) = \begin{cases} X = a_{x0} + a_{x1}t + a_{x2}t^2 \\ Y = a_{Y0} + a_{Y1}t + a_{Y2}t^2 \\ Z = a_{Z0} + a_{Z1}t + a_{Z2}t^2 \end{cases}$, the shortest distance means solving $\min_t \left\| (X-x)^2 + (Y-y)^2 + (Z-z)^2 \right\|$. Assume $G(t) = (X-x)^2 + (Y-y)^2 + (Z-z)^2$, then $G(t)$ is a quartic equation, and it is always positive. To get min $G(t)$, we need to calculate where $G_{\prime(t)=0}$. The number of solutions for $G_{\prime(t)=0}$ may be from 3 to 1. If there is more than one solution, the one that gets the minimum $G(t)$ is the final solution. The projected point is $F(t)$.

4.6 Solve the system

We applied a linear least squares solver for the FEM system and a non-linear least squares solver for the energy minimization system. The constraint setting for both solvers is the same.

4.6.1 Set constraints for solver

A constraint point is meant to be immovable (hard constraint) or slightly movable (soft constraint) after solving. The partial derivative determines the direction and magnitude of converging in regression problems. For hard constraints, the partial derivatives of the constraint points should be set to 0. For soft constraints, the partial derivatives should be set to a large value. If the partial derivative is 0, that means the variable will not change after regression. If the partial derivative is a large value, the variable will change slightly, because a small change will affect the whole system significantly.

For FEM, the stiffness matrix is actually the partial derivatives of deformation. Therefore, the constraint setting for FEM is to set a 0 or a large value for the constraint points in K matrix. For the thin shell energy minimizing method, it is to set a 0 or a large value for the constraint points in the Jacobian matrix.

4.6.2 Solve FEM system

After we implemented the co-rotational stiffness matrix, the relationship between force and deformation is no longer linear. However, the relationship between force and the deformed shape is still linear: recall the formula after co-rotational stiffness: $F = RK(R^{-1}x - x_o)$. Since the values of R, k and x_o are known, the formula can be written as:

$$F = RKR^{-1}x - RKx_o = Ax - b \tag{39}$$

Now $min||Ax - b||_2$ is a typical linear least squares problem. For the whole system, the final matrix of A will be a large matrix with a lot of "0"s. In our case, the template has around 7000 vertices and over 2000 tetrahedron elements. The size of the stiffness matrix K is more than 24000x21000. Therefore, a sparse matrix data structure is required. We used the Eigen library for the sparse matrix operations. Eigen is a C++ library designed for linear algebra and numerical solvers[43].

Since we adopted the co-rotational stiffness method, the result will take several iterations to get a good shape. The whole procedure now becomes:

1. Build initial stiffness matrix K for every tetrahedron.
2. Solve the system and get the deformed shape X.
3. Extract the rotation matrix R between the initial shape X_o and deformed shape X for each tetrahedron.
4. Build the co-rotated stiffness matrix and solve for the min force: $\min \left\| RK(R^{-1}x - x_o) \right\|_2$
5. Iterate steps 3 and 4 for N times.

Ideally, the more iterations, the more accurate the result is. The figures 17 and 18 show how the results differ between 4, 16 and 256 iterations. The red template takes 4 iterations and 23 seconds. The orange template takes 8 iterations and 52 seconds. The yellow template takes 16 iterations and 85 seconds. The green template takes 256 iterations and 1588 seconds. From the results, we can draw the conclusion that the 16 iterations one maintains a balance between performance and efficiency, so we chose 16 iterations for our system.

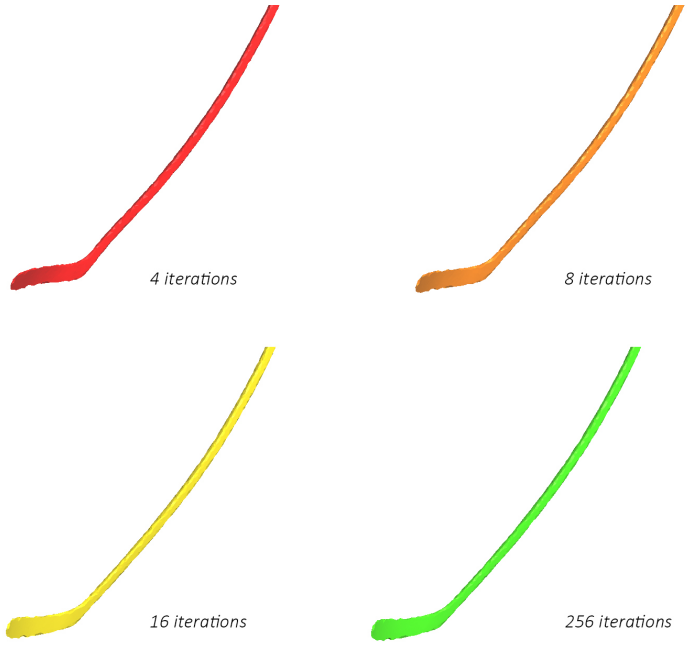

Figure 17: The results of 4 different number of iterations

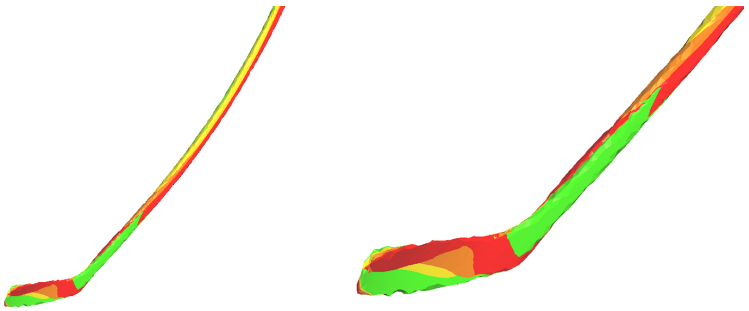

Figure 18: The overlapping of results of different iterations

4.6.3 Solve the discrete shell energy system

To solve the discrete shell energy system is to solve $min||E||_2$, where

$$E = E_e + E_A + E_B = K_L \sum_e (1 - \frac{e}{e_o})^2 e_o + K_A \sum_A (1 - \frac{A}{A_o})^2 A_o + k_B \sum_e (\theta_e - \theta_{e_o})^2 e_o / h_{e_o} \tag{40}$$

It is a non-linear problem. We used the ceres-solver to solve our system. The ceres-solver is an open source C++ library for modeling and solving large, complicated optimization problems[44]. The ceres solver supports automatic derivative. However, to set the constraints, we calculated the Jacobian matrix and change the variables where they are constraint points to the corresponding hard constraint or soft constraint value. The system is then solved by ceres-solver using trust region method.

Chapter 5

Results and conclusion

We tested our system on different sticks and shot styles. In this section, several frames of the results of 4 shots with 2 different sticks and 2 different shot types are provided. We evaluate the quality of our method by overlapping the deformed shape on the reconstructed point cloud. We also apply a commonly used quantifying method to evaluate the results and compare them with the results of MOCAP system.

5.1 Results for the 4 shots

We tested our pipeline on 2 shot styles, the Slap Shot, and the Wrist Shot. The players hit the puck from a small distance in a wrist shot, mostly using the strength from the lower body. The slap shot is the hardest shot. Players raise the hockey stick to shoulder high or higher and hit on the ground slightly behind the puck. The hockey stick is bent like a spring, storing the energy, and then hits the puck. We also tested with 2 different templates. The results promisingly reproduced the bending shape from captured images. Since our goal is to reconstruct the deformation of hockey sticks in 3D space for better observations, we evaluate the quality of the results by overlapping the deformed template with the reconstructed point cloud. We also followed a current state of art quantifying method to evaluate the bending accuracy.

Here are the wrist shots with 2 different stick template. Stick 1 is a bit shorter than stick 2.

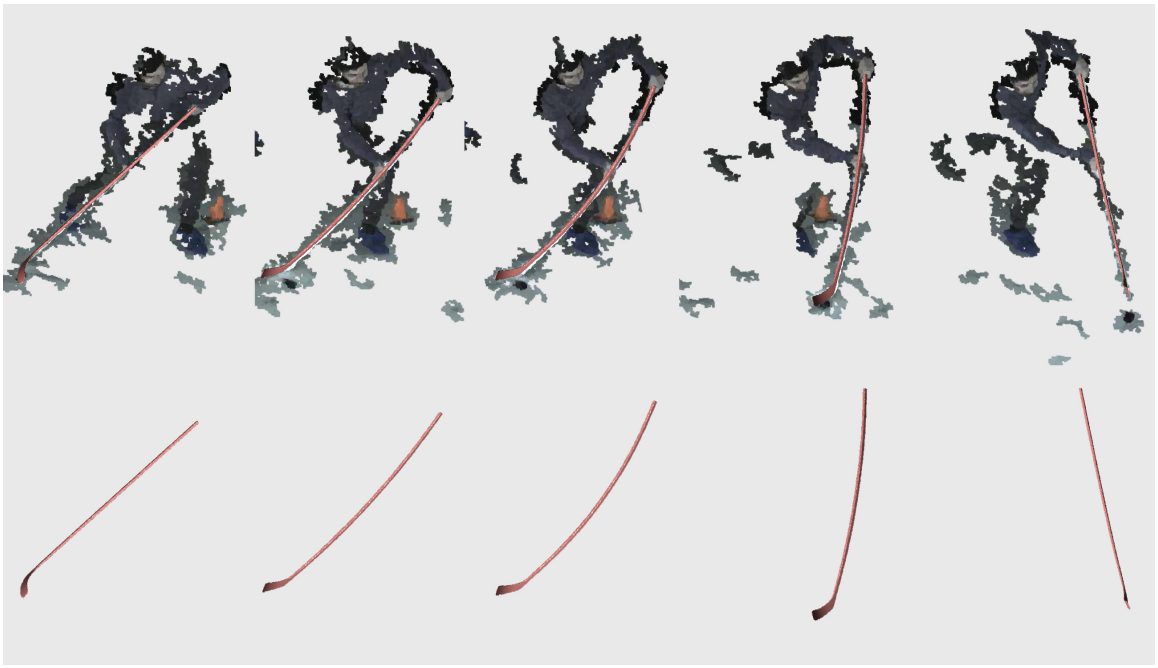

Figure 19: A few frames of a wrist shot with stick 1

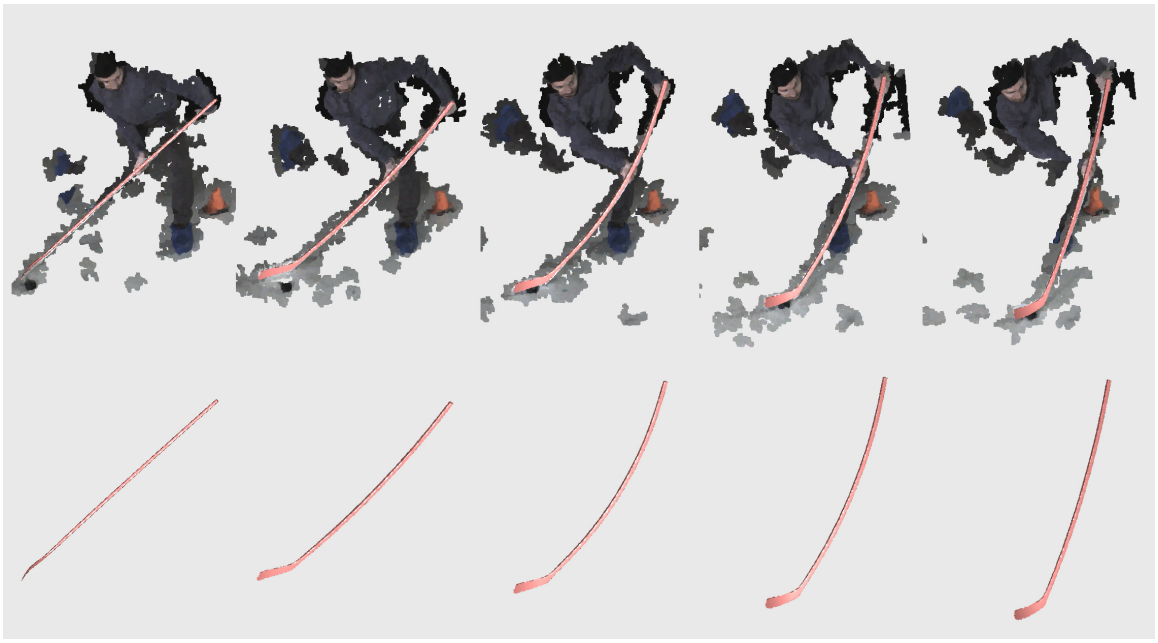

Figure 20: A few frames of a wrist shot with stick 2

The wrist shot style is not as severe as the slap shot. However, we can still observe the nice bending shape from the deformed template.

In a slap shot, the player hits the puck at a very fast speed, and the stick bends more than the wrist shot. Therefore, it is more challenging for the reconstruction of slap shots. We tested on several slap shots, and the results are all satisfying. Here are the two slap shots with the two stick templates.

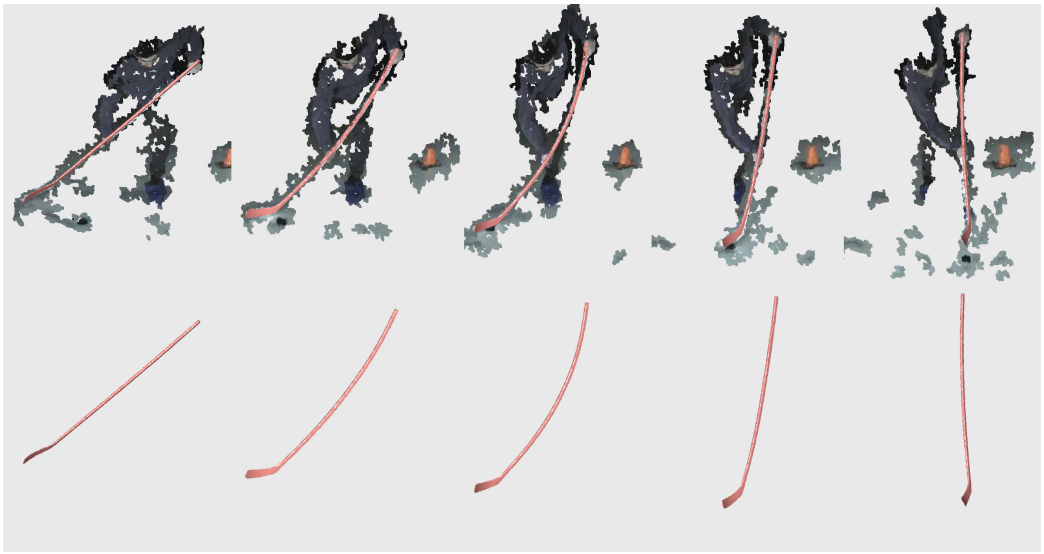

Figure 21: A few frames of a slap shot with stick 1

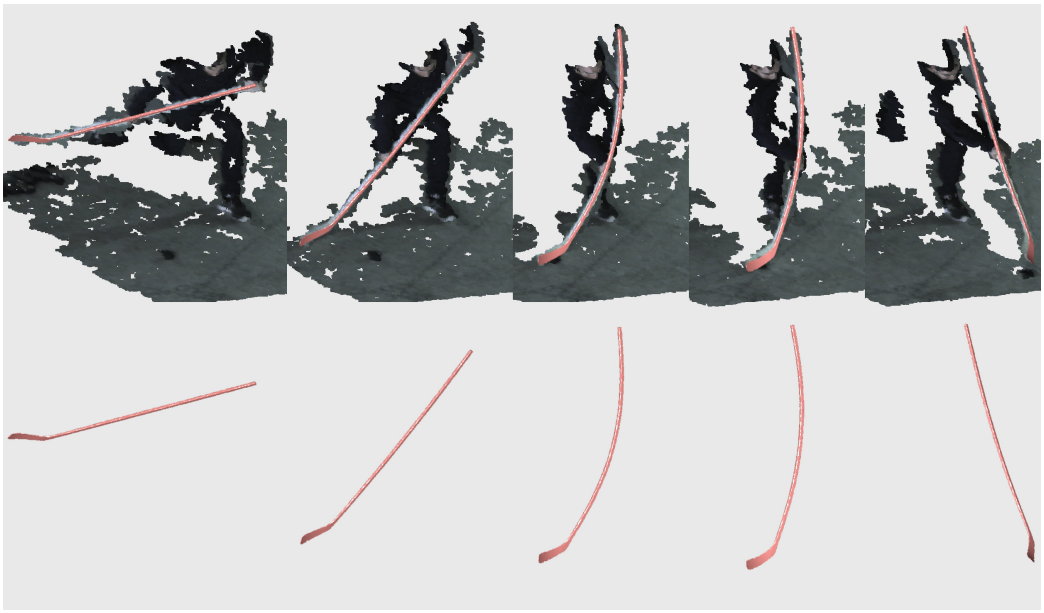

Figure 22: A few frames of a slap shot with stick 2

As shown in the figures, the deformed shape matches closely to the point cloud. Even the deformation caused by inertia is captured in the last frames.

5.1.1 Quantify Bending Accuracy

In the industry, a commonly used method to quantify the bending of the hockey stick is to attach markers to the stick and capture the trace of the markers using a motion capture (MOCAP) system. The figure 23 shows how we attached the markers on the hockey stick. We set virtual markers on the template according to the real hockey stick and markers. The markers are divided into 3 groups A, B, and C. The three markers of each group can define a surface, and their normal vectors are used to calculate the angles. We computed 2 angles: the angle between groups A and B and the angle between groups A and C.

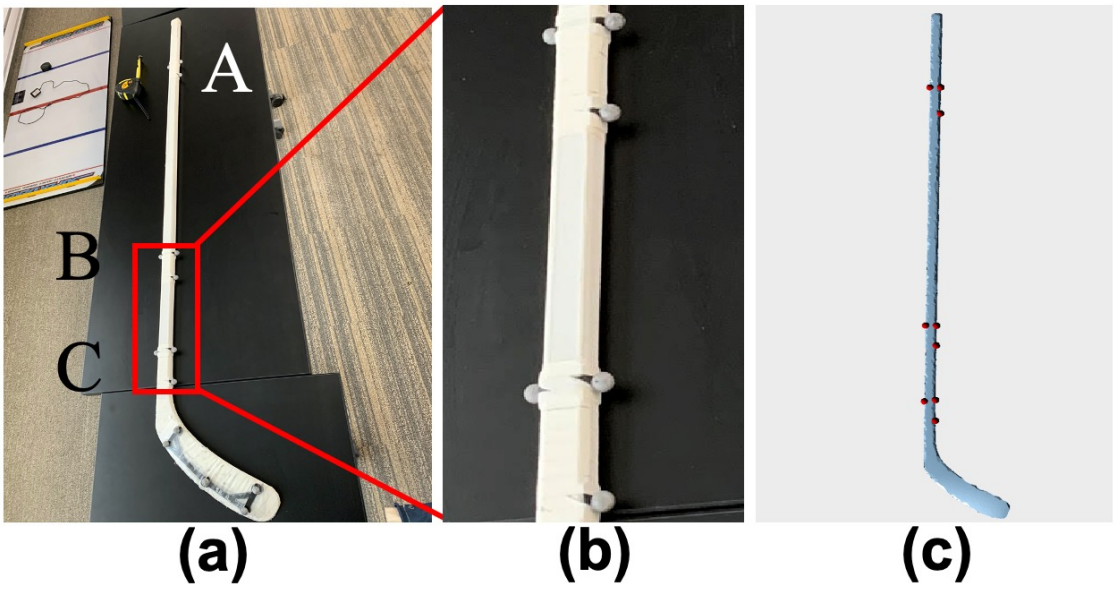

Figure 23: The setup for markers

And here is the comparison with the data from MOCAP system and our system with Finite Element Method (FEM) and the Discrete Shell Energy Method (DSEM)

	Slap Shot	**Wrist Shot**
Max Angle between A and B MOCAP	21.80	20.00
Max Angle between A and B FEM	23.65	22.05
Max Angle between A and B DSEM	22.90	22.44
Diff between MOCAP and FEM	1.85	2.05
Diff between MOCAP and DSEM	1.10	2.44
Diff between FEM and DSEM	0.75	0.39
Max Angle between A and C MOCAP	26.20	23.15
Max Angle between A and C FEM	28.35	26.65
Max Angle between A and C DSEM	28.59	26.35
Diff between MOCAP and FEM	2.15	3.50
Diff between MOCAP and DSEM	2.39	3.20
Diff between FEM and DSEM	0.24	0.30

Table 1: Quantitative comparison of the maximum bending angle between the Finite Element Method, the energy minimizing method, and the MOCAP system

The results are presented in table 1. Note that some errors may come from the set up of the markers. The markers on the real stick are small balls stuck on the stick, while the virtual markers are selected from the vertices. The expected difference in the maximum bending angle is about 4 degrees.

5.1.2 Comparison between two physics-based methods

In general, both FEM and the discrete shell energy method generate convincing results. On the one hand, the finite element with linear tetrahedrons is more cumbersome because it requires re-meshing the template and generating a tetrahedralization of the 3D model while the Discrete Shell Energy Method operates directly on the triangle meshes. On the other hand, the thin shell energy minimizing method is more difficult for solving since it is a non-linear method. Although the FEM model is also non-linear after we applied the co-rotational stiffness technique, it can be reduced to iteratively solving linear systems that converges in only few iterations. From the performance perspective, the linear finite

element takes the volume into account while the thin shell energy minimizing method only calculates the energy on triangle surfaces. We want to find how this difference affects the final results. From table 1, we can tell that the two methods give very similar results. Here are three frames of the two methods. Both of them generate good results, and the overlapping suggests that their results of the deformed shape are almost the same.

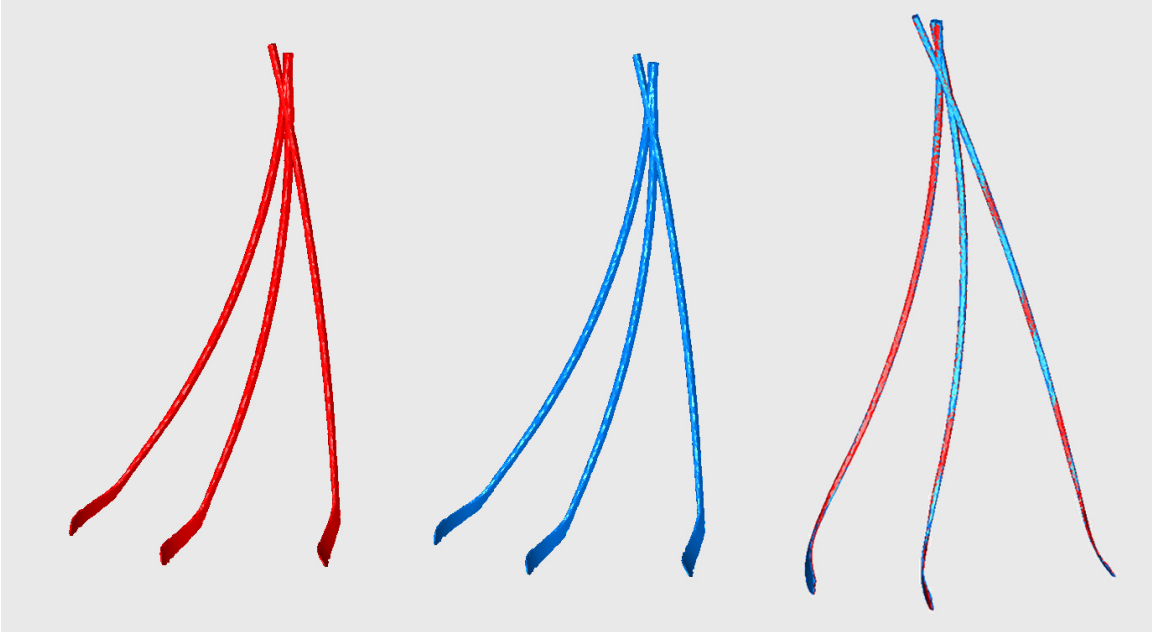

Figure 24: The comparison of the two physics-based methods. The red one is the energy minimizing method, and the blue one is the Finite Element Method.

A main difference between the two methods is their efficiency. The run-time for FEM is mostly affected by the number of iterations for co-rotational stiffness operation. The average run-time for each iteration is around 5.4 seconds. For the 16 iterations that we chose, it takes an average of 87 seconds. For the energy minimizing method, the run-time depends on the threshold of convergence. It is not stable since the convergence of each frame is not under the same situation. The number of iterations for different frames varies from around 600 to over 10000, and the run-time ranges from 11 seconds to 260 seconds. The overall run-time for the energy minimizing method is shorter than that for the finite element method. Here is the table of their run-times on the same sequence:

	The Finite Element Method	The Discrete Shell Energy Method
Min Run-time	85.7s	11.49s
Max Run-time	88.5s	263.45s
Average Run-time	87.3s	41.38

Table 2: Run-time comparison between FEM and the Discrete Shell Energy method

5.2 Conclusions

In this thesis, we implement a module that consists of constraint setting, physics-based models, and solvers to reconstruct the deformation of hockey sticks. The co-rotational Finite Element Method and the Discrete Shells Method are adopted to build our physics-based models, and they are compared. The two physics-based models generate very close results, and the results are promising compared to the original videos and the reconstructed point cloud. The main difference is that the run-time of co-rotational FEM for every frame is stable while run-time of the Discrete Shell Energy Minimizing Method varies between different frames.

A main limitation is the deformation of the blade part is hard to reproduce because the blade part only bends when it hits the puck. As a result, the reconstruction always contains the noise from the puck and the noise is hard to remove. In the future, we are planning to solve this problem by detecting and separating the puck in the scene. Another limitation is that currently we are selecting the constraint points manually using some principles. This could be a source of error and future works need to be done to find a method that automatically select the constraint points.

Bibliography

[1] W. contributors, "Stress–strain curve." https://en.wikipedia.org/wiki/Stress-strain_curve/.

[2] J. Worobets, J. Fairbairn, and D. Stefanyshyn, "The influence of shaft stiffness on potential energy and puck speed during wrist and slap shots in ice hockey," Sports Engineering, vol. 9, no. 4, pp. 191–200, 2006.

[3] B. T. Kays and L. V. Smith, "Effect of ice hockey stick stiffness on performance," Sports Engineering, vol. 20, no. 4, pp. 245–254, 2017.

[4] M. Swarén, Q. Söhnlein, T. Stöggl, and G. Björklund, "Using 3d motion capture to analyse ice hockey shooting technique on ice," in icSPORTS2019, Vienna, Austria, 20-21 September, 2019, pp. 204–208, SciTePress, 2019.

[5] K. Mendhurwar, G. Handa, L. Zhu, S. Mudur, E. Beauchesne, M. LeVangie, A. Hallihan, A. Javadtalab, and T. Popa, "A system for acquisition and modelling of ice-hockey stick shape deformation from player shot videos," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 3893–3900, June 2020.

[6] S. Parashar, D. Pizarro, A. Bartoli, and T. Collins, "As-rigid-as-possible volumetric shape-from-template," in Proceedings of the IEEE International Conference on Computer Vision, pp. 891–899, 2015.

[7] Z. Li, A. Heyden, and M. Oskarsson, "Template based human pose and shape estimation from a single rgb-d image," in 8th International Conference on Pattern Recognition Applications and Methods, ICPRAM 2019, pp. 574–581, SciTePress, 2019.

[8] D. Russell and L. Hunt, "Spring constants for hockey sticks," The Physics Teacher (submitted draft). Retrieved from: www. acs. psu. edu/drussell/publications/russell-hunt-tpt-formatted. pdf (Accessed 17 January 2014), 2009.

[9] D. Bradley, T. Boubekeur, and W. Heidrich, "Accurate multi-view reconstruction using robust binocular stereo and surface meshing," in 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8, IEEE, 2008.

[10] D. Bradley, T. Popa, A. Sheffer, W. Heidrich, and T. Boubekeur, "Markerless garment capture," in ACM SIGGRAPH 2008 papers, pp. 1–9, 2008.

[11] M. H. Sadd, Elasticity: theory, applications, and numerics. Academic Press, 2009.

[12] D. R. Askeland, P. P. Phulé, W. J. Wright, and D. Bhattacharya, "The science and engineering of materials," 2003.

[13] A. J. M. Spencer, Continuum mechanics. Courier Corporation, 2004.

[14] R. Blickhan, "The spring-mass model for running and hopping," Journal of biomechanics, vol. 22, no. 11-12, pp. 1217–1227, 1989.

[15] J. Jiang, B. Sheng, P. Li, L. Ma, X. Tong, and E. Wu, "Real-time hair simulation with heptadiagonal decomposition on mass spring system," Graphical Models, p. 101077, 2020.

[16] T. I. Vassilev, "Mass-spring cloth simulation with shape matching," in Proceedings of the 18th International Conference on Computer Systems and Technologies, pp. 257–264, 2017.

[17] S. S. Rao, The finite element method in engineering. Butterworth-heinemann, 2017.

[18] A. Hrennikoff, "Solution of problems of elasticity by the framework method," J. appl. Mech., 1941.

[19] R. Courant et al., "Variational methods for the solution of problems of equilibrium and vibrations," Lecture notes in pure and applied mathematics, pp. 1–1, 1994.

[20] J. N. Reddy, Introduction to the finite element method. McGraw-Hill Education, 2019.

[21] "The finite element method: Its basis and fundamentals," in The Finite Element Method: its Basis and Fundamentals (Seventh Edition) (O. Zienkiewicz, R. Taylor, and J. Zhu, eds.), Oxford: Butterworth-Heinemann, seventh edition ed., 2013.

[22] M. Mahran, A. ELsabbagh, and H. Negm, "A comparison between different finite elements for elastic and aero-elastic analyses," Journal of Advanced Research, vol. 8, no. 6, pp. 635 – 648, 2017.

[23] S. Benzley, E. Perry, K. Merkley, B. Clark, and G. Sjaardema, "A comparison of all hexagonal and all tetrahedral finite element meshes for elastic and elasto-plastic analysis," Proceedings, 4th International Meshing Roundtable, vol. 17, 01 1995.

[24] T. J. Willmore, "Surfaces in conformal geometry," Annals of Global Analysis and Geometry, vol. 18, no. 3-4, pp. 255–264, 2000.

[25] J. C. Butcher and N. Goodwin, Numerical methods for ordinary differential equations, vol. 2. Wiley Online Library, 2008.

[26] H. Von Der Mosel, "Minimizing the elastic energy of knots," Asymptotic Analysis, vol. 18, no. 1, 2, pp. 49–65, 1998.

[27] J. Bonet and R. D. Wood, Nonlinear continuum mechanics for finite element analysis. Cambridge university press, 1997.

[28] B. Smith, F. D. Goes, and T. Kim, "Stable neo-hookean flesh simulation," ACM Transactions on Graphics (TOG), vol. 37, no. 2, pp. 1–15, 2018.

[29] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler, "Stable real-time deformations," in Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 49–54, 2002.

[30] M. Müller and M. H. Gross, "Interactive virtual materials.," in Graphics interface, vol. 2004, pp. 239–246, 2004.

[31] O. Etzmuss, M. Keckeisen, and W. Strasser, "A fast finite element solution for cloth modelling," in 11th Pacific Conference onComputer Graphics and Applications, 2003. Proceedings., pp. 244–251, 2003.

[32] M. Wardetzky, M. Bergou, D. Harmon, D. Zorin, and E. Grinspun, "Discrete quadratic curvature energies," Computer Aided Geometric Design, vol. 24, no. 8-9, pp. 499–518, 2007.

[33] E. Grinspun, A. N. Hirani, M. Desbrun, and P. Schröder, "Discrete shells," in Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 62–67, Citeseer, 2003.

[34] M. Merriman, A List of Writings Relating to the Method of Least Squares: With Historical and Critical Notes, vol. 4. Academy, 1877.

[35] A. C. Rencher, Methods of multivariate analysis, vol. 492. John Wiley & Sons, 2003.

[36] J. Nocedal and S. Wright, Numerical optimization. Springer Science & Business Media, 2006.

[37] D. C. Sorensen, "Newton's method with a model trust region modification," SIAM Journal on Numerical Analysis, vol. 19, no. 2, pp. 409–426, 1982.

[38] Z.-J. Shi, "Convergence of line search methods for unconstrained optimization," Applied Mathematics and Computation, vol. 157, no. 2, pp. 393–405, 2004.

[39] "Graphite is an experimental 3d modeler, built in top of the geogram programming library.." http://alice.loria.fr/software/graphite/doc/html/index.html.

[40] H. Si, "Tetgen, a delaunay-based quality tetrahedral mesh generator," ACM Transactions on Mathematical Software (TOMS), vol. 41, no. 2, pp. 1–36, 2015.

[41] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.

[42] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," International journal of computer vision, vol. 13, no. 2, pp. 119–152, 1994.

[43] B. Jacob, G. Guennebaud, et al., "Eigen is a c++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms." http://eigen.tuxfamily.org/.

[44] S. Agarwal, K. Mierle, and Others, "Ceres solver." http://ceres-solver.org.