A Pipeline for Modelling of Ice-Hockey Stick Shape Deformation Using Actual Shot Video

Gaurav Handa

A Thesis

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Computer Science at

Concordia University

Montréal, Québec, Canada

October 2020

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Gaurav Handa**

Entitled: **A Pipeline for Modelling of Ice-Hockey Stick Shape Deformation Using Actual Shot Video**

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 Committee:

______________________ Chair
Dr. Charalambos Poullis

______________________ Examiner
Dr. Marta Kersten-Oertel

______________________ Supervisor
Dr. Tiberiu Popa

Approved by ______________________
Dr. Lata Narayan, Chair
Department of Computer Science

__________ 2020 ______________________
Amir Asif, Dean
Faculty of Engineering and Computer Science

Abstract

A Pipeline for Modelling of Ice-Hockey Stick Shape Deformation Using Actual Shot Video

Gaurav Handa

In Ice-Hockey, performance of the player's shots depends on their skill level, body strength as well as stick's construction and stiffness. In fact, research suggests that one of the primary reasons that the elite players generate much faster shots is their ability to flex their hockey stick. Thus, reconstructing the deformable 3D shape of the stick during the course of a player shot has important applications in performance analysis of ice-hockey stick.

We present a new, low cost, portable system to acquire videos of a player shot and to automatically reconstruct the stick shape's deformation in 3D. This thesis is a sub-part and contributes towards the ultimate goal of the pipeline in many different ways. First, designing a mobile stereo-vision setup and its calibration, capturing a lot of data acquisitions with different players shooting in different styles. Second, developing a two step pruning methodology to prune structurally thin and fast moving ice-hockey stick from noisy reconstructed point cloud in 3D. Third, automating the process of initial rigid alignment of the stick template in the noisy reconstruction. Forth, reducing the effect of noise by using medial axis approximation approach and suppressing the hand occlusion effect on the final template bending by a curve fitting approach. This pipeline is also robust against different ice-hockey sticks along with different players, shooting at different styles.

Acknowledgments

I wish to express my greatest gratitude to my supervisor, Dr. Tiberiu Popa, for continuous support, inspiration and guidance during my journey on this research work. Special thanks to Marc LeVangie, Aiden Hallihan from CCM-Hockey, Montreal, for being extremely supportive during all my necessary experiments at their facility. I also want to thank Leixiao Zhu & Etienne Beauchesne and my lab mates Shakil Ahmmed & Sourush for all the healthy discussions and useful reviews. I also wish to acknowledge my friend Prashansa Gupta for her great support without which I would not have been able to complete this work. I also want to thank my family & especially remember my father, Gopal Krishan whose values continue to inspire me throughout this journey.

Contents

List of Figures

Chapter 1

Introduction

Ice-Hockey is a very popular team sport, characterized by high speed skating and rapid movement of players. It is an equipment heavy sport where players use a variety of gears to play. One of the most important piece of equipment of this sport is the ice-hockey stick. The global market for ice-hockey sticks was estimated at 240 million USD in 2018 and is expected to reach 320 million USD by the end of 2025, growing at a compound annual growth rate (CAGR) of 3.6 percent during 2019-2025 [5]. A lot of technical advances are happening towards designing equipment for ice hockey [6].

Performance of the player's shots in ice hockey depends on their skill level, body strength as well as sticks construction and stiffness. Rules of the game stipulate on the dimension of the shaft and blade; however, there is no restriction on the material composition of sticks [7]. As such, stick manufacturers have focused on making use of composite materials which allows them to modify the sticks' mechanical characteristics and evaluate improvements via shot mechanics [8]. This evaluation is dependent on variables like peak bending of the stick, peak puck velocity, stick to ground angles, peak angular deflection of the shaft, and many more. In fact, research suggests that one of the primary reasons that the elite players generate much faster shots is their ability to flex their hockey stick [9]. Thus, capturing bending of the stick is an important aspect of performance analysis of the ice-hockey stick.

But capturing this bend in the ice-hockey stick is a very challenging problem. Ice-hockey is a very fast sport in terms of motion where a player shoots at very high speed. The average puck

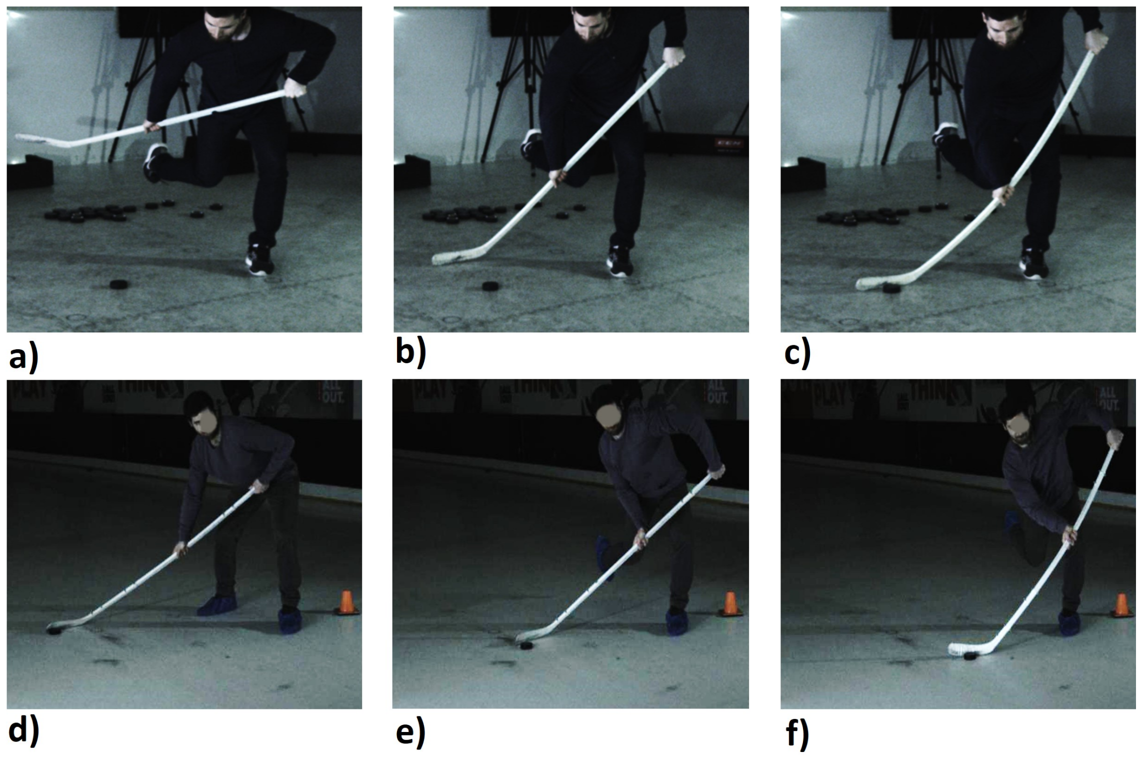

Figure 1.1: A visualization of slap and wrist shots in Ice-Hockey: Figure a)-c) depict the stages in the slap shot and Figure d)-f) shows stages in the wrist shot of ice-hockey. The potential energy stored in the bend of the stick transforms into kinetic energy of puck during shooting.

velocity for an adult group is close to 97 km/h [10] and the bending in the stick happens for a very short time. To properly capture this swift bending in the stick we need a high frequency acquisition setup. With experimental testing we found that a vision based acquisition setup needed to have a minimum frame rate of 250 frames-per-second (fps) to meticulously capture the bending details.

Apart from motion, structure of the ice-hockey stick poses other big challenges. An ice-hockey stick is structurally a thin object which generally doesn't have very good texture for 3D-reconstruction. Additionally, the camera system has to be placed at a safe distance from fast moving pucks which increases the camera capture volume. The thin structure and high capture volume adds to the complexity of the stick segmentation. Thus, recognizing the stick in the scene without using any added constraints of extra markers is a very challenging problem and one of the motivation of this work.

After segmentation, Occlusion presents the next big challenge in recovering the true bend shape of the stick. At any point of time, a part of the stick is at least occluded by the hand gloves of the player. If not taken into account, this occlusion many a times results into undesired bumps in the final bending (see Figure-4.26).

We propose a complete software pipeline along with hardware setup to capture and reconstruct bending of an ice-hockey stick. We present a new, low cost, portable system to acquire videos of a player shot and to automatically reconstruct the deformation in the 3D shape of the stick using a template based approach [4]. For acquisition, we use portable high-speed stereo video cameras and reconstruct the stick in 3D along with its deformation during various hockey shots. To perform the 3D reconstruction, we use a template based approach also known as the Shape From Template (SFT) [11] approach. We use a shape template of the ice-hockey stick as provided by the manufacturer. This template is then deformed using motion priors or physics-based principles to match the current observation. For each captured frame, we deformed a mesh based ice-hockey stick template to match the actual bending happening during the shot. Automatic initial rigid-alignment of this known stick-template in the reconstructed 3D scene is another big challenge to address.

This thesis is a sub-part of the proposed pipeline which has many other components apart from those discussed here. Section 4.2 explains different components of the complete pipeline, while Section 4.3 summarizes the exact component wise contribution of this thesis towards the proposed solution. As a part of the pipeline, this thesis majorly addresses the following challenges,

(1) Segmentation of the ice-hockey stick both in two and three dimensional spaces

(2) Automatic rigid alignment of stick template in reconstructed scene by means of point feature matching

(3) Removal of the undesirable bending effect due to noise in reconstruction and hand occlusion using a medial axis approach

The remaining thesis is organized as: Chapter 2 review related literature work and Chapter 3 summarizes the theoretical background needed to understand the various components of the pipeline. Chapter 4 explains in detail, the overall pipeline and the technical contribution of this thesis. Chapter 5 showcases the qualitative and quantitative results. Chapter 6 includes the concluding remarks and scope for future work.

Chapter 2

Related Work

Various research studies on the game of Ice-Hockey are available in the literature looking at different aspects of the game. A recent study in pose estimation and action recognition in the game of Ice-Hockey is presented by [12]. For the purposes of stick performance analysis, however, 3D reconstruction/representation of the hockey stick is imperative, and not much work has been reported on this problem. Optical motion capture studies from the perspective of 3D representation are available in the literature for analyzing various shots employed in the game of Ice-Hockey. Some of these studies have been carried out in a simulated environment i.e., by making use of the synthetic ice [13], [14], [15], [16]. A recent study has also been carried out on real ice with professional hockey players [17] to study the 3D kinematics of various shots played in Ice-Hockey. A few of the major drawbacks in using these high-end, high accuracy motion capture equipment are (i) the intrusive nature of the markers, (ii) high cost of equipment and its set-up, and more importantly (iii) non-portable setup.

3D reconstruction from images and video has been at the core of many of the research problems for several decades. A lot of literature is available in both computer vision and computer graphics domains for 3D reconstruction, and similar representations often appear in both the domains. This is owing to the fact that vision researchers are solving the shape recovery of real objects problem, whereas graphics researchers are simulating the deformations of virtual ones [18]. The approaches can be broadly classified into structure from motion (SFM) and template based reconstruction or shape from template (SFT) [19].

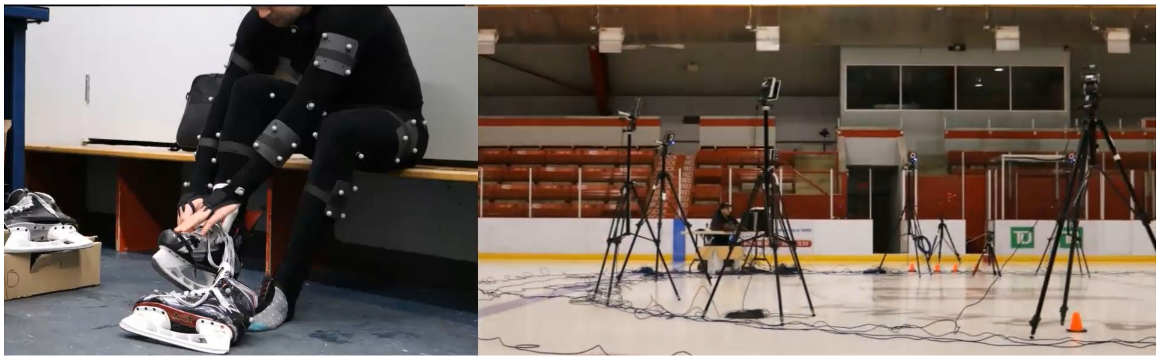

Figure 2.1: An illustration of state-of-art infra-red based motion capture system. Reflective markers are placed on the player and equipment, which were analyzed by the camera system to study the kinematics of the sport [Source: Ice Hockey Research Group, Mcgill University].

In SFM approaches, a set of points are tracked on a set of images, which then are used to reconstruct the objects. A study on recovering the shape and motion of a single rigid object is presented by [20]. An assumption of object shape being a linear combination of some basis shapes has been made to recover shapes of deformable objects in [21], [22]. Stick movement in Ice-Hockey is very fast and such rapid motion leads to large frame-to-frame differences, which makes tracking challenging, especially for highly deformable objects [23].

In SFT or template-based reconstruction approaches, a shape template of the non-rigid object is available a priori. This template is then deformed using motion priors or physics-based principles to match the current observation. This has been demonstrated to estimate the shapes of human faces by [24]. SFT approaches can be further subclassified based on input data, namely monocular, RGB-D and stereo image sequences.

Monocular Data: Wangg et al [25] have proposed a method for reconstructing 3D face expressions from monocular video sequences using a 3D face template mesh. Parashar et al [26] have proposed volumetric SFT to reconstruct an object's surface and interior deformation using a single image and a 3D object template. Perriollat et al [27] have proposed reconstruction of sheet-like inextensible surfaces using a single image taken from a camera with known intrinsic parameters and a template. Alldieck et al [28] have proposed a method to create personalized realistic 3D human models from a monocular video sequence and a parametric SMPL body model. Zuffi et al [29] have proposed a method to obtain 3D textured animal model, given a set of images of the animal annotated with landmarks and silhouettes. Deep learning has also been employed in several research works for 3D object reconstruction based on a single image. A detailed review on state-of-the-art deep learning methods for image-based 3D object reconstruction is also available in the literature [30]. The difficulty in using this monocular approach for our problem is that during a shot, the orientation of the hockey stick keeps changing very rapidly and a single view is unable to provide all the information needed for deformation modeling.

RGB-D Data: Microsoft introduced Kinect in 2010 and ever since affordable RGB-D devices like Intel RealSense, Primesense Carmine, Google Tango, Occipital have made RGB-D database creation very easy. Many foundational research problems have been revisited and rethought to make best use of the new capabilities of RGB-D cameras, and a detailed survey is available in [31]. Li et al [32] have proposed a method to reconstruct the 3D human body model from a single RGB-D image and a parametric body model. Tao Yu et al [33] have proposed a new real-time system that combines volumetric dynamic reconstruction with data driven template fitting to simultaneously reconstruct detailed geometry, non-rigid motion and the inner human body shape from a single depth camera. Raoul de Charette et. al. [34] have proposed a method to reconstruct arbitrary 3D revolving objects (in context of live pottery making) that handles deformation as well as occlusion using one or more depth sensors. The primary problem with these acquisition systems is that the speed of capture is very low and a fast-moving hockey stick is difficult to capture using current state-of-the-art RGB-D sensors.

We opted for a stereo setup consisting of two high-speed global shutter cameras with hardware temporal synchronization. The system is capable of capturing at a very high frequency typically around 275 frames per second, which is sufficient for high speed motion involved in the ice-hockey shots. Capturing the scene with two different views help to cope up with the changing ice-hockey stick orientation problem with single view setup. Some of the other major features of our setup is its portability and economic feasibility. The system is assembled on a moving structure which can be moved locally and can easily be dissembled-assembled to take out to remote places. Also, the setup is relatively cheaper as compared to other sophisticated and expensive motion capture systems.

Chapter 3

Technical Background

In this section, we discuss the theoretical concepts that form the foundation of the overall process described in this thesis, at a high-level. Firstly, in Section 3.1 we explain the concepts behind point clouds. We discuss idea behind the binocular stereo vision setup and its calibration. Then, we introduce the idea of a point cloud descriptor which helps us perform automatic initial rigid alignment of the stick template in the scene. In Section 3.2, we discuss the theoretical background of optical flow particularly, dense optical flow which helps us in pruning point clouds. After that, we describe the basics of different algorithms that we use at multiple occasions in this thesis namely, RANdom Sample Consensus (RANSAC) in Section 3.3 and Iterative Closest Point (ICP) algorithm in Section 3.4. Lastly, we discuss the L1-Medial skeleton extraction process in Section 3.5.

3.1 Point Cloud

For a long time, just 2D representation (images) of the real world provide solutions to complex problems like remote sensing, microscopic imaging and many more. In today's modern world, adding the third dimension (depth) in the representation facilitates us to deliver more advanced applications like autonomous vehicles, 3D printing and much more. With the advancement of hardware technologies, there exist different techniques to generate the 3D representation of any object. We can use modern 3D sensor technologies like projected light sensors (E.g: Kinnect) or Time-of-flight sensors (E.g: LiDAR) to generate the 3D representation of any object. We can even use multiple 2D images from different views of the same object to generate its 3D representation (stereo vision). We can store and process this 3D representation in different modes like RGB-D images or point clouds. In our pipeline, we use the stereo vision principle to generate the 3D representation of the scene in the form of a point cloud data.

A Point cloud is defined as a set of data points that represent object or space. Each point includes atleast the X, Y and Z geometrical coordinates of an actual point. By means of point clouds we can combine a large number of single spatial measurements into one dataset which represent a complete object. Apart from geometrical coordinates a point cloud may also store a number of other parameters like color, point-normal etc. There are a number of formats in which we can represent a point cloud like .obj, .pcd, .epts etc.

Applications of point clouds can be seen in different domains. In the manufacturing industry, point cloud are used to create 3D CAD model of manufactured parts for quality inspection. In the automobile industry, a great amount of research has already been done to use point cloud based depth perception for driver-less cars. Point clouds are also used for rendering and animation applications.

3.1.1 Point cloud generation: Binocular Stereo-vision

The technique of estimating 3D information of a scene by using images from two or more viewpoints is known as stereo-vision. If only two images corresponding to different viewpoints of the same scene are involved then it is called binocular stereo-vision. We have used two identical cameras setup (see Section 4.1) separated by a baseline distance of approximately 1m center-to-center. Let's now discuss the geometry of this binocular stereo setup using a simple model as shown in Figure 3.2 and see how we can retrieve depth information from it.

In this model, we have two identical cameras separated by a distance b in x-direction only called baseline distance. In this simple model, we assume image planes to be co-planar. Any 3D point in the scene is located at different 2D positions on both the image planes. These points on different images that are projection of same real world point are called conjugate pair. The difference (displacement) between the locations of this point on the image planes is of much importance and known as the disparity. In simpler terms, disparity may be defined as the difference between the

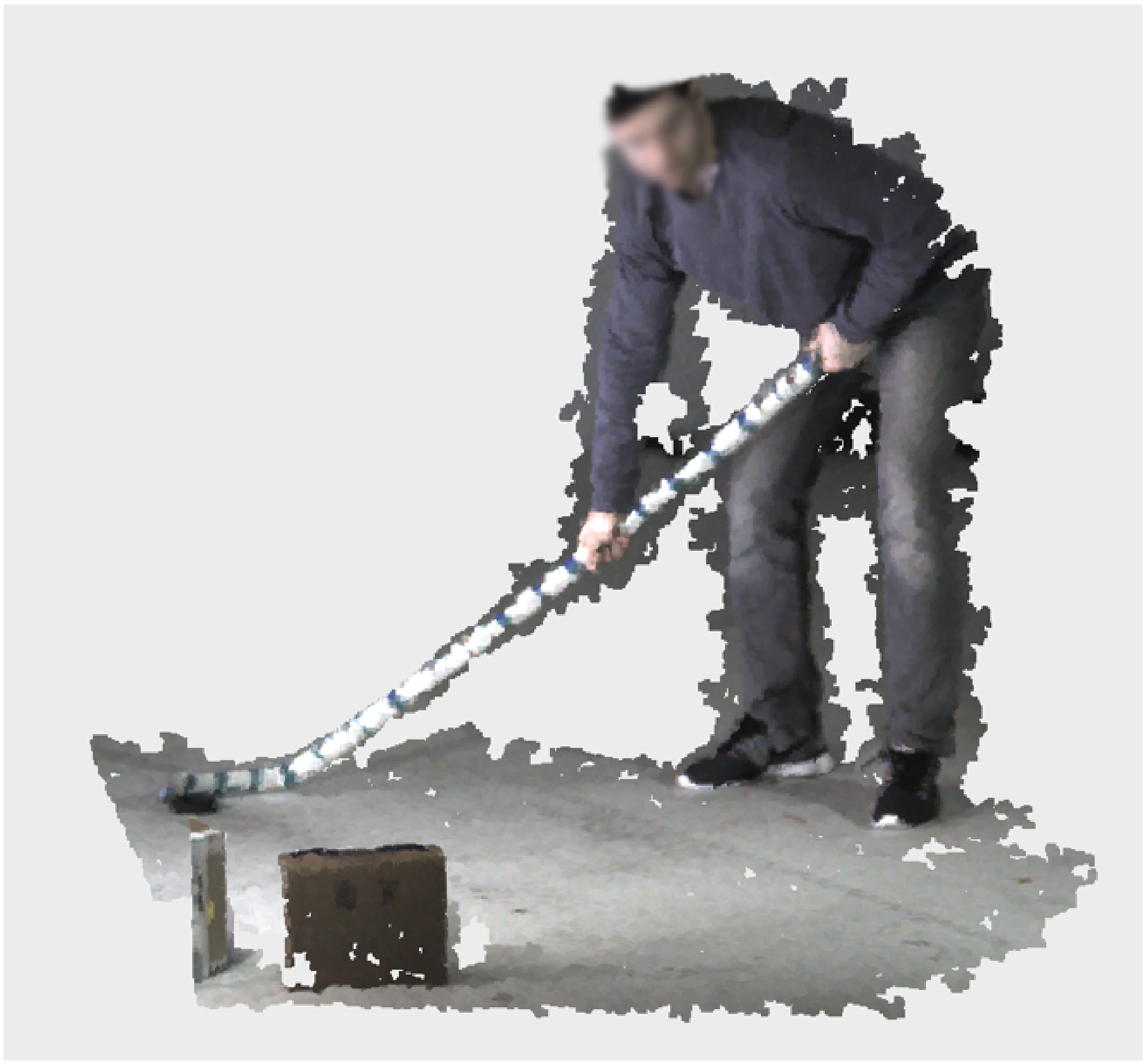

Figure 3.1: Visualizing point cloud: An example of point cloud representing an an actual object

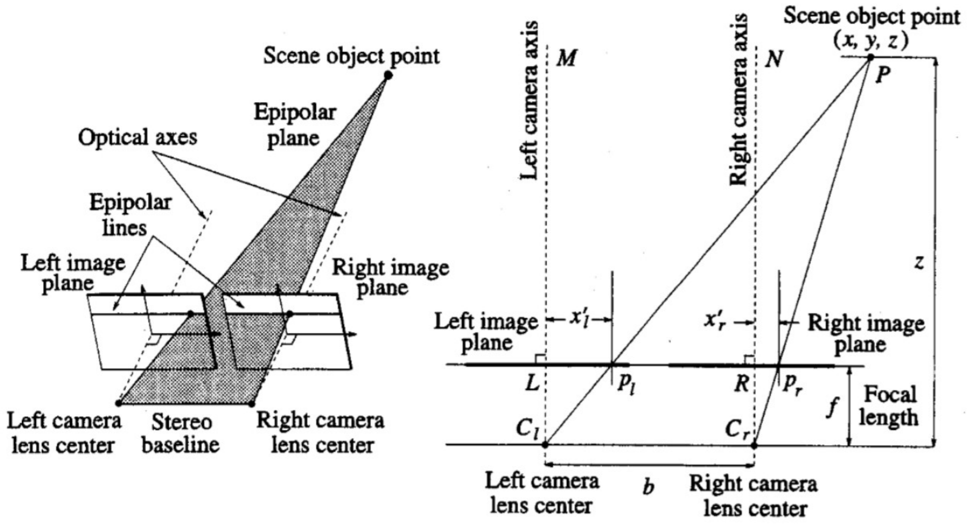

Figure 3.2: A simple model representing the binocular stereo vision geometry (Source [1])

conjugate pairs when we superimpose these images. An imaginary plane passing through the point and the centers of both the cameras is known as epipolar plane. Let us consider a 3D point P with geometrical coordinates as (x, y, z) where z is the depth of the point in real world. Let this point form point $P_{\mathrm{l}}(x_{\mathrm{l}}, y_{\mathrm{l}})$ and $P_{\mathrm{r}}(x_{\mathrm{r}}, y_{\mathrm{r}})$ as conjugate pair and $x_l^{'}$, $x_r^{'}$ as disparity values in left and right images respectively. Then from the similarity of triangles we can say,

$$\frac{z}{b} = \frac{f}{x_l^{'} - x_r^{'}} \tag{1}$$

which can be written as,

$$z = b\frac{f}{x_l^{'} - x_r^{'}} \tag{2}$$

Where, f is the focal length of cameras and b is the disparity. Thus, we have observed that the depth of real world points can be retrieved with the knowledge of conjugate pairs and their disparity. Here, we made an assumption that we know the conjugate pairs in the stereo images. The problem of detecting conjugate pairs is known as correspondence problem and there exists various methods, correlation-based, feature based to name a few, to solve it.

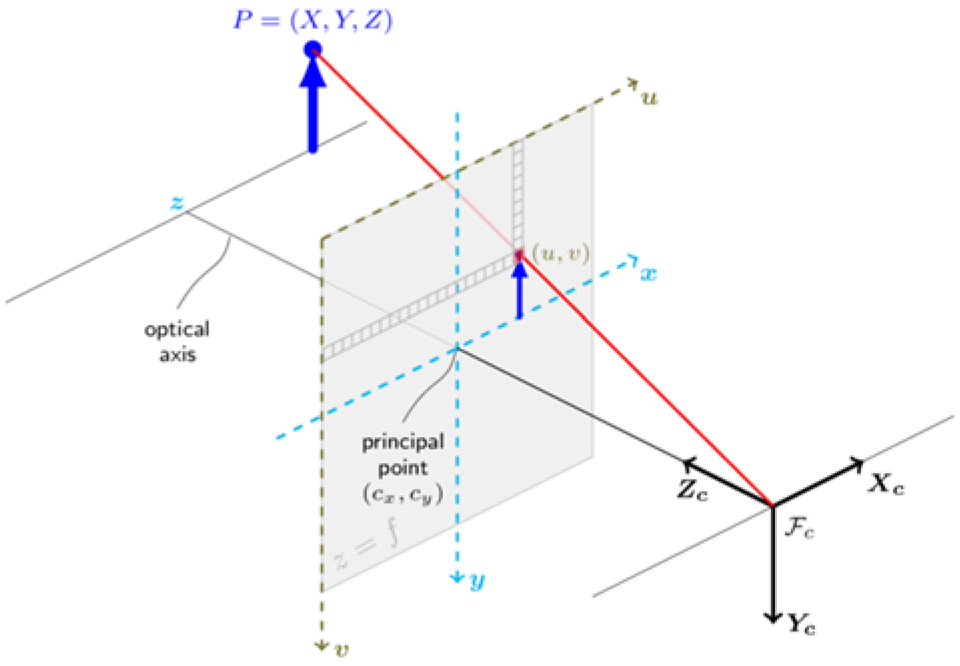

Figure 3.3: Pinhole model of camera (courtesy: openCV)

3.1.2 Camera calibration

Camera calibration is defined as the process of finding intrinsic and extrinsic parameters. These camera parameters are also called as internal and external parameters respectively. Intrinsic parameters are those camera parameters which deal with the camera's internal characteristics like focal length, image center, distortion and skew. Extrinsic parameters deal with the camera's external characteristics like, camera's position and orientation with respect to the world.

Camera calibration is one of the vital steps in any 2D or 3D stereo vision application. There are different techniques available for camera calibration but one of the most popular one is photogrammetric calibration. In this technique, calibration is performed by using a calibration object like, calibration bars or checkerboard pattern.

Let us consider a pinhole camera model as described in Figure 3.3. Under this model, a scene view is generated by using perspective transformation of 3D points on image plane. A point object with coordinates (X, Y, Z) is projected at location (u, v) on an image screen by using the following perspective transformation:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3}$$

Where (c_x, c_y) is principal point representing the center of the image, f_x and f_y are focal lengths with pixel units and s is the scaling factor. The joint matrix [**R**,**t**] is known as the extrinsic matrix, where **R** corresponds to rotation while **t** corresponds to translation. This extrinsic matrix describes the rigid motion of the moving object in front of the still camera. In simpler words, the extrinsic matrix transforms the coordinates of an actual point to a fixed coordinate system with respect to the camera. We can rewrite the transformation as below,

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t \tag{4}$$

$$x^{'} = x/z \tag{5}$$

$$y^{'} = y/z \tag{6}$$

$$u = f_x x^{'} + c_x \tag{7}$$

$$v = f_y y^{'} + c_y \tag{8}$$

In the real world, there are many distortions which influence this model. Like radial distortion introduced by defects in manufacturing of lens and distortion arising from camera sensor not completely orthogonal to lens axis. To accommodate such distortions, we have to extend the above discussed camera model. The model mentioned below accommodates radial distortion and slight

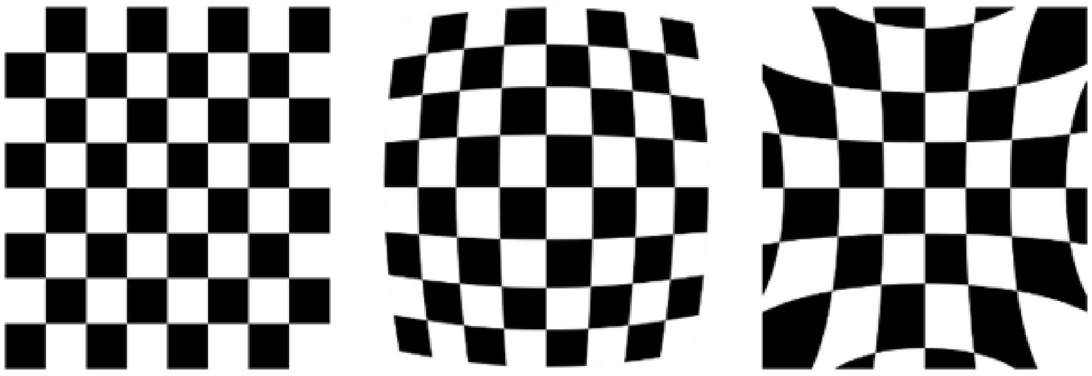

Figure 3.4: Left to right, example showing no-distortion, positive radial and negative radial distortion respectively (courtesy: openCV).

tangent distortion:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t \tag{9}$$

$$x^{'} = x/z \tag{10}$$

$$y^{'} = y/z \tag{11}$$

$$x^{"} = x^{'} \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_1 x^{'} y^{'} + p_2(r^2 + 2x^{'2}) \tag{12}$$

$$y^{"} = y^{'} \frac{1 + k_1 r^2 + k_2 r^4 + k_3 r^6}{1 + k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_2 x^{'} y^{'} + p_1(r^2 + 2y^{'2}) \tag{13}$$

where

$$r^2 = x^{'2} + y^{'2} \tag{14}$$

$$u = f_x x^{"} + c_x \tag{15}$$

$$v = f_y y^{"} + c_y \tag{16}$$

Where, k_1, k_2, k_3, k_4, k_5, k_6 are the radial distortion coefficients while p_1, p_2 are tangential distortion coefficient.

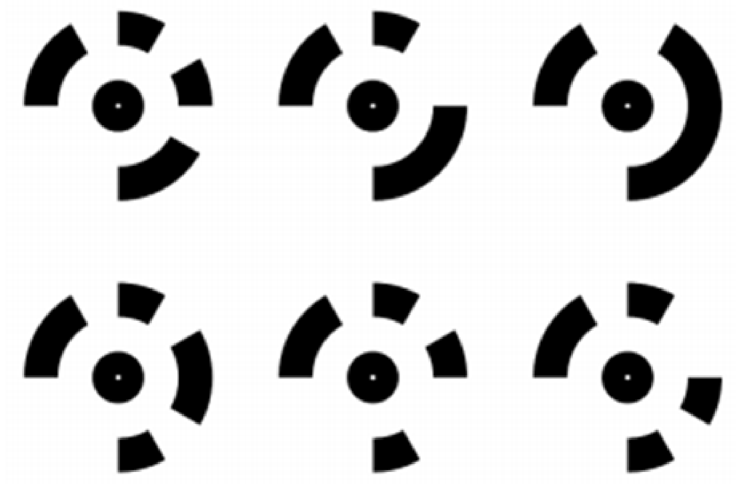

Figure 3.5: Coded circular scale bars used for calibration. The distances between the centers of sequentially arranged circular figures are known which helps calibrate the setup.

The calibration and 3D-reconstruction part of the pipeline is implemented in the Agisoft's Photoscan API. In order to support high accurate measurement of the 3D data we use coded circular scale bars shown in Figure 3.5. Before capturing any acquisition, we calibrate the cameras. Once the cameras are calibrated, we can then capture any number of acquisitions without disturbing the setup. For the purpose of calibration, we used the setup mentioned in the Figure 3.6. In addition to coded circular scale bars, we used a colored calibration cloth to increase the correspondence between left and right images for better results.

We optimize parameters k1, k2, k3, p1, p2 to accommodate distortion effects inside the Photoscan API. Sometimes, because of varied lightening effects markers are not automatically captured, for which we have to mark them as seen in Figure 3.7. After performing the calibration step we export the calculated camera positions and the parameters evaluated in an .xml file for later use.

After calibration the next step in Photoscan is to reconstruct each frame captured in the acquisition. By using the already saved calibration data we then reconstruct each frame of the acquisition and save the results. The reconstruction of every frame is saved in the form of point cloud dataset.

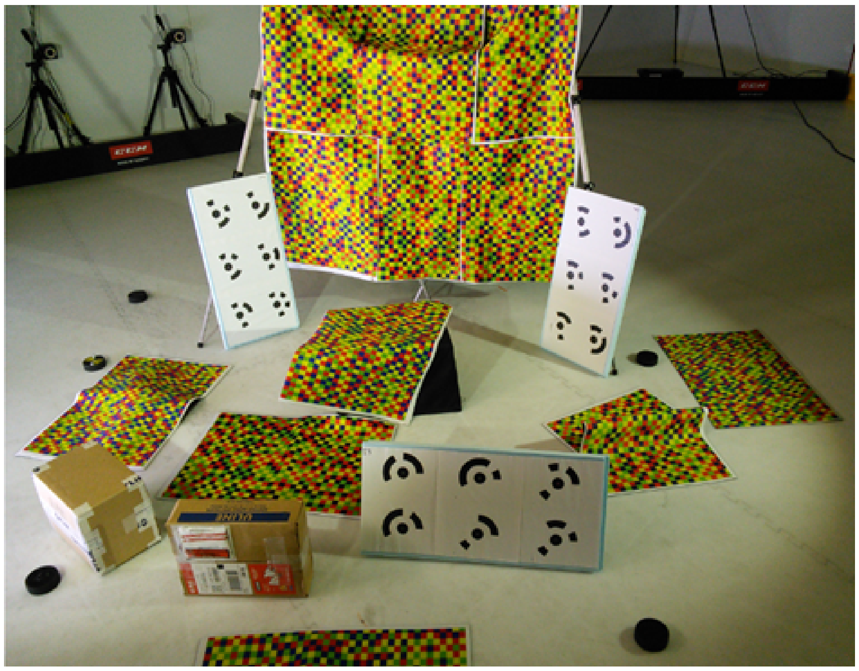

Figure 3.6: An example setup we used for one of the acquisition. For calibration, we use coded circular scale bars and calibration cloth. The color calibration cloth helps to increase the correspondence between the stereo images and give better results.

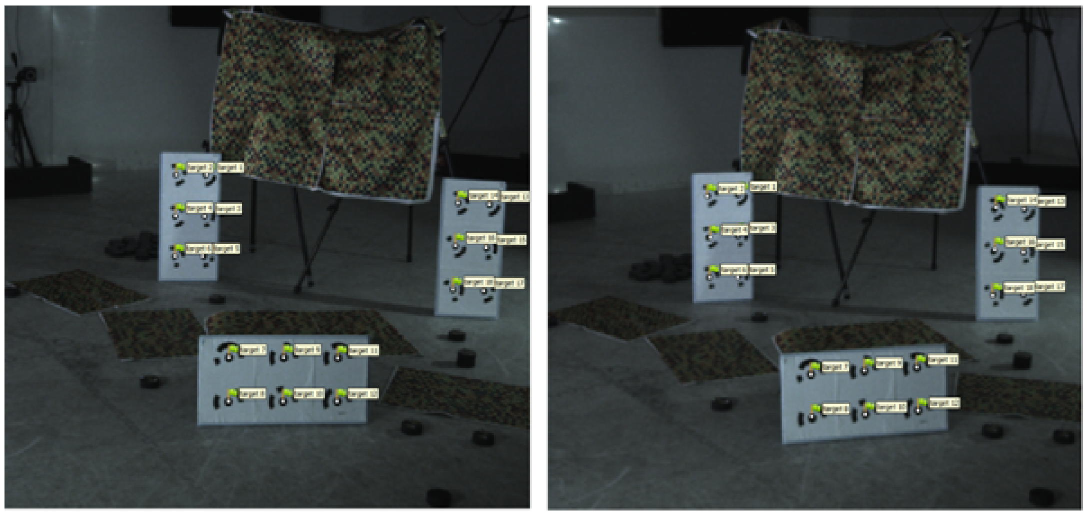

Figure 3.7: Images show the detected circular rings on the calibration bars. Left image belongs to the left cameras and right image belongs to the right camera of the stereo setup. These circular coded bars are automatically detected under Agisoft API.

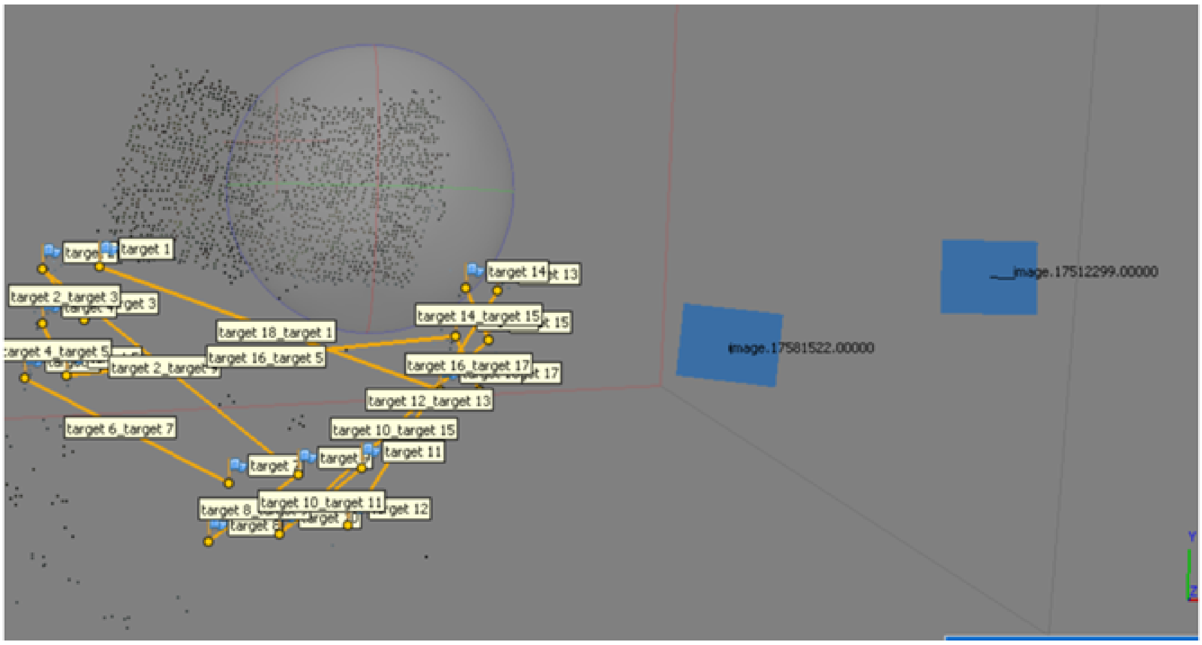

Figure 3.8: Image showcases the determined relative camera locations (blue boxes) and initial tie points generated after the calibration step inside Agisoft API.

3.1.3 Point cloud descriptor: Point Feature Histogram

In any 3D point cloud based pipeline, one of the crucial steps is point feature extraction as we can't understand the scene just with the points' 3D coordinates. Point cloud descriptors are compact point features which could help better understand the geometry represented by the point cloud. These descriptors should be invariant to translation, rotation and scale. One of the prominent uses of point cloud descriptors is seen in multi-view point cloud registration problem. In this pipeline, we use point cloud descriptors to find the initial rigid alignment of the stick template with the pruned point cloud.

There exists different types of point features which can be broadly categorised as local, global and hybrid, depending upon the technique used to calculate them. Surface normal and curvature are the most commonly used local descriptors as they are calculated by using the closest neighbours of a point. There exist different robust feature descriptors like moment invariants and spherical harmonic invariants etc.

There are a plethora of techniques to quantify the 3D features in multi-value form like spin

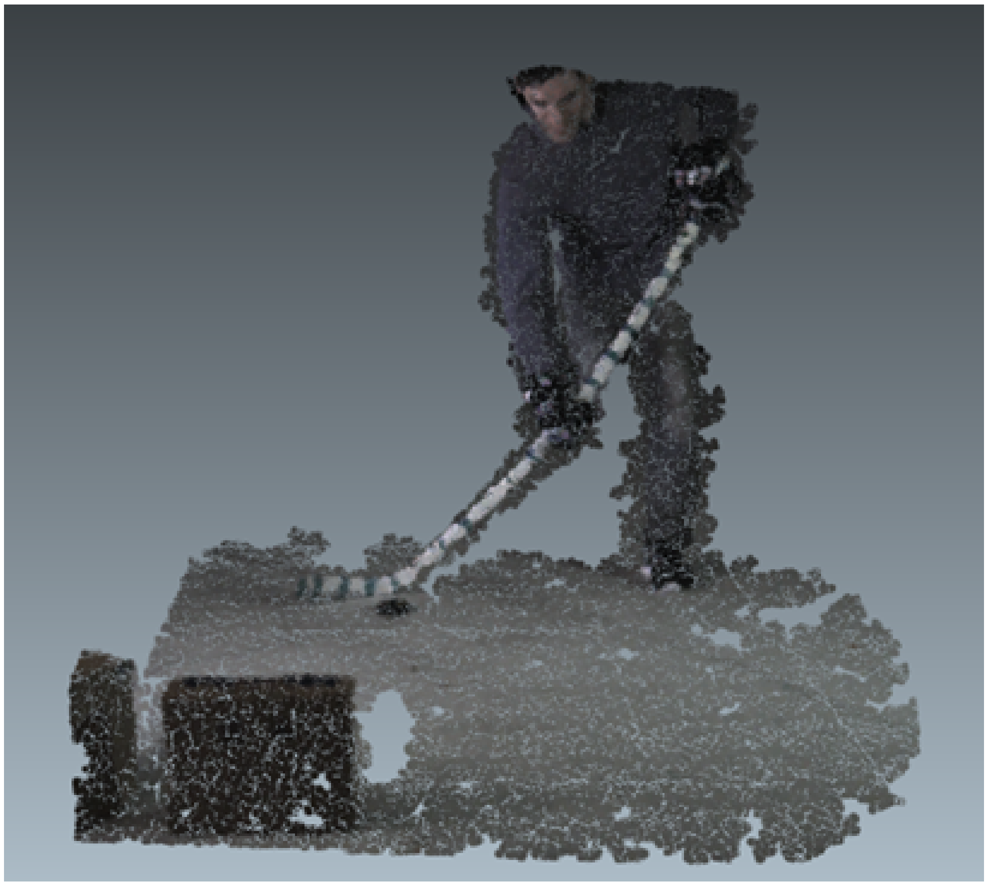

Figure 3.9: An example of the generated point cloud

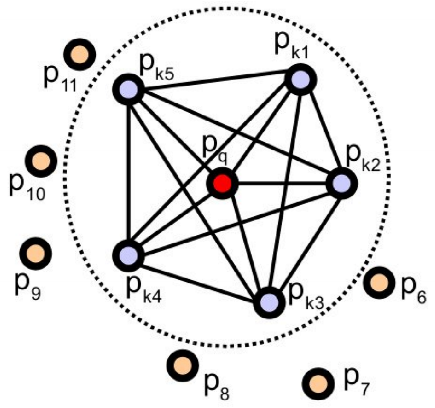

Figure 3.10: An illustration of influence region diagram for a Point Feature Histogram. The query point (red) and all of its k-neighbours (blue) are shown, source: [2]

image signature and curvature based histogram to name a few. They are created from basic local features (surface nomals, curvatures) and establish relationship between them by binning them in a histogram. The Point feature histogram (PFH) [2] is an extension of this research line of multi-value feature descriptors.

In PFH, we compute the local point feature histogram for every point in the dataset. PFH quantifies the geometrical properties of the neighbouring points in the influence region of the query point (see Figure 3.10) into density and pose invariant multi-value feature descriptor. In order to generate PFH we have to first calculate the surface normal at each point of the point cloud. Now, to compute the PFH at point p, we first consider a spherical influence zone of radius r with the center at query point p. We then list all k-neighborhood points within radius r. For every pair p_i and p_j $(i \neq j)$ and their respective normals n_i and n_j, we will select source p_s and target p_t, where source is the point which will have smaller angle between the respective normal and the line connecting both points. Once the source is decided then a Darboux frame is generated with origin at the source point: $u = n_\text{s}, v = (p_\text{t} - p_\text{s}) \times u, w = u \times v$. The four geometrical features, as mentioned below, are calculated and categorized using 16-bin histogram. Where each bin at index idx captures the percentage of the point pairs in the k-neighbourhood which have their features in the interval defined by idx.

$$idx = \sum_{i=1}^{i \leq 4} step(s_{\mathrm{i}}, f_{\mathrm{i}}) \cdot 2^{i-1} \tag{17}$$

Here $step(s, f)$ is a step function with value 1 if $f > s$ and 0 otherwise with,

$$f_1 = v \cdot n_{\mathrm{t}} \tag{18}$$

$$f_2 = \|p_{\mathrm{t}} - p_{\mathrm{s}}\| \tag{19}$$

$$f_3 = u \cdot (p_{\mathrm{t}} - p_{\mathrm{s}})/f_2 \tag{20}$$

$$f_4 = \arctan(w \cdot n_{\mathrm{t}}, u \cdot n_{\mathrm{t}}) \tag{21}$$

When s_1,s_3,s_4 are fixed as 0 and s_2 as r, then each feature of a point pair in the neighbourhood of query point is divided in two categories and encodes the percentage of pairs with same category for all the features in the histogram. In our pipeline, we use a modified version of PFH known as Fast Point Feature Histogram (FPFH) [35], which is the optimised implementation of PFH for real time application.

3.2 Optical Flow

Motion is a vital part of our visual experience. This motion is caused by a relative movement between the observer and the scene. Optical flow deals with the structure/pattern of motion of objects in the scene. Optical flow is defined as the distribution of relative motion, pixel displacement vectors within an image which belongs to a sequence of frames. For 2D images, optical flow is generally represented in the form of 2D-vector fields. Each of these 2D-vectors represents a displacement vector, showing that movement of a point in two consecutive image frames. Optical flow has a large number of different applications, like it is used for object segmentation, robotic navigation, stereo disparity measurements and many more. We have used optical flow for object (ice-hockey stick) segmentation in this pipeline.

There are two major types of optical flow namely sparse optical flow and dense optical flow.

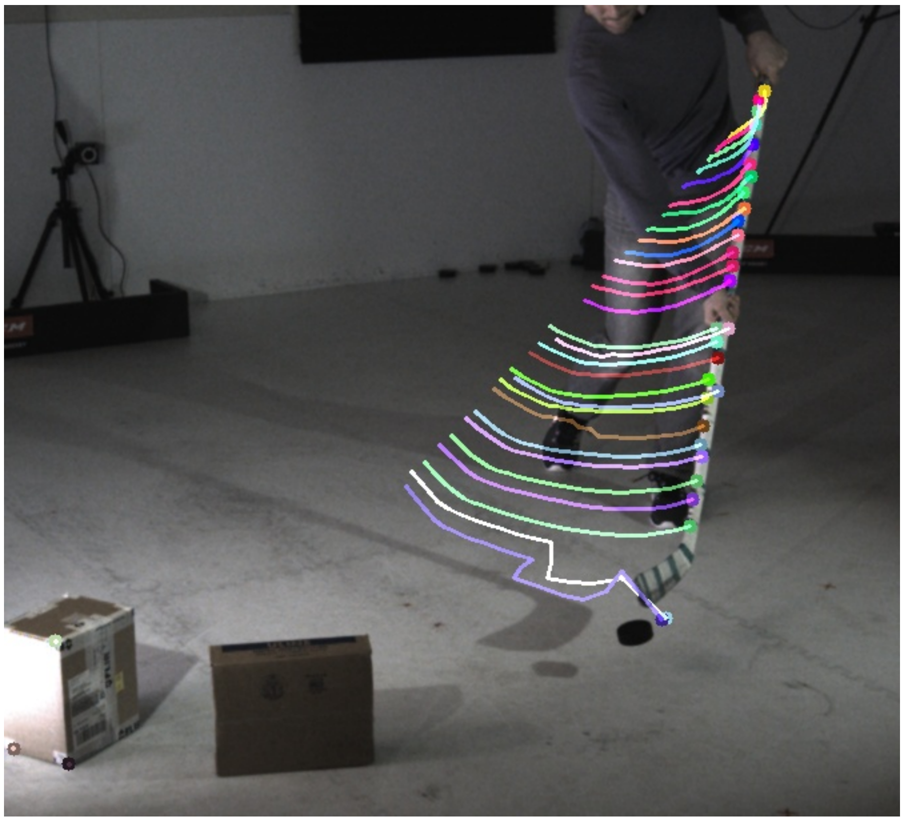

Figure 3.11: An illustration of object motion tracking using movement of pixels in the sequence of frames. Colored paths in the figure depict the motion of respective point location on the stick from starting frame to current frame using Lucas Kanade sparse optical flow algorithm [3].

Sparse optical flow computes optical flow for a sparse feature set like corners etc. while dense optical flow computes optical flow for all the points in the image. Sparse optical flow is fast but less accurate while dense optical flow is slow but more accurate. There are different implementations available for both sparse and dense optical flows. The Lucas-Kanade algorithm [3] is broadly used for sparse optical flow estimation while Gunner Farneback algorithm [36] is for dense optical flow calculation.

3.2.1 Dense Optical Flow: Farneback's two frame optical flow

In Farneback's algorithm [36], a dense optical flow is calculated using two consective frames. The calculation of optical flow is based upon the polynomial expansion of a neighbourhood of pixel. Each pixel's neighbourhood in both frames is approximated by quadratic polynomials $f(x)$, where A, b, c are symmetric matrix, vector and scalar respectively.

$$f(x) \sim x^T A x + b^T x + c \tag{22}$$

Lets consider a polynomial signal $f_1(x)$

$$f_1(x) = x^T A_1 x + {b_1}^T x + c_1 \tag{23}$$

It goes through an ideal translation d and results into a new signal $f_2(x)$.

$$\begin{aligned} f_2(x) &= f_1(x-d) \\ &= (x-d)^T A_1 (x-d) + b_1^T (x-d) + c_1 \\ &= x^T A_1 x + (b_1 - 2A_1 d)^T x + d^T A_1 d - b_1^T d + c_1 \\ &= x^T A_2 x + b_2^T x + c_2 \end{aligned} \tag{24}$$

After equating coefficients of $f_2(x)$ with $f_1(x)$ we get,

$$A_2 = A_1 \tag{25}$$

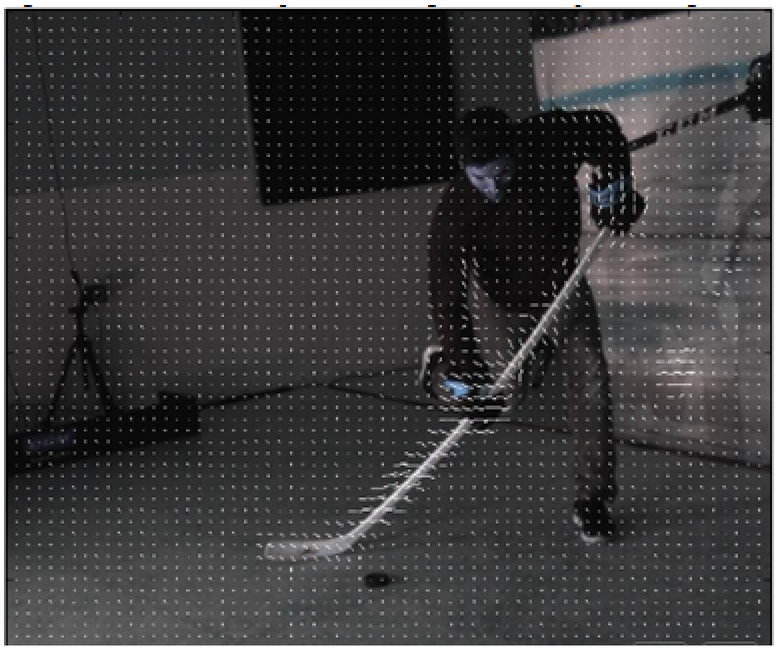

Figure 3.12: Showcasing the vector representation (white lines) of the resultant Dense optical flow calculated using the Farneback's two frame optical flow algorithm with OpenCV. The displacement of pixels observed by the optical flow is vividly visible around the ice hockey stick which is the dominant moving object in the selected image pair.

$$b_2 = b_1 - 2A_1 d \tag{26}$$

$$c_2 = d^T A_1 d - b_1^T d + c_1 \tag{27}$$

One major observation from this equation comparison is that if A_1 is non-singular then we can solve for translation value d as following.

$$d = -\frac{1}{2} A_1^{-1}(b_2 - b_1) \tag{28}$$

Thus, we can determine the displacement vector for every pixel using consecutive pairs of frames. In the practical implementation, the optical flow is determined in both the axis of the image and Figure 3.12 showcases the vector representation of the resultant optical flow displacement vector.

3.3 Random Sample Consensus (RANSAC)

The Estimation of parameters of a mathematical model from given data points has been an active area of research. This parameter estimation becomes quite challenging in the presence of a high number of outliers in the input data. The RANdom Sample Consensus (RANSAC) algorithm [37] is one of the popular choice for such scenarios. We use RANSAC based approach for initial rigid alignment of the stick template in the starting frame with the noisy reconstructed point clouds.

RANSAC is an iterative algorithm which helps estimate mathematical model parameters from a given set of observations that contain outliers. This algorithm assumes that there exist inliers (may be small) whose distribution can be mapped using some model parameters. RANSAC uses a random sampling iterative approach to finalize the parameters. This algorithm builds up different models using different sampling in each case and data elements in the dataset used to find the optimal solution. First, a random selection of the minimum required points is made from the dataset and model parameters are computed from the selection. Second, using the estimated parameter a check is performed with all points in the dataset to validate their fitting to the model based upon some defined threshold value.

Algorithm 1: Random Sample Consensus

```
1 Randomly select the required minimum number of points to calculate the parameters of the
    model.
2 Evaluate the model parameters using sampled points.
3 Using the estimated model, determine the number of points from complete dataset who fall
    under the predefined tolerance value.
4 If inliers fraction is above a predefined threshold, re-calculate the model parameters using
    all identified inliers and quit.
5 Else, repeat step 1 through 4 (maximum of $N$ times).
```

3.4 Iterative Closest Point (ICP)

ICP [38], sometimes also known as Iterative Corresponding Point algorithm is one of the major methods to geometrically align 3D models when an initial relative pose is known $a\ priori$. A major use of ICP could be seen in the 3D model generation using 3D scanning techniques. These scanners produce scans of the object from one view at a time, which need to be registered together to construct a complete 3D model. In this pipeline, we used ICP to refine the initial rigid alignment of stick template to point cloud and propagate the bend template from previous the frame to the next frame. Because of its popularity and widespread use today there exist many different variants of the ICP depending upon different techniques of selection, matching of points and minimization of error. An outline of the generic flow of ICP includes the following steps:

(1) Initialize the parameters for registration **R**,**t** and assign error metric to infinity.

(2) For every point in the source point cloud, find the corresponding closest point in the target point cloud.

(3) Calculate the value of **R**,**t** for the given correspondence from step 2.

(4) Apply the transformation calculated from last step to the source point cloud.

(5) Calculate the error in the registration of source and target point clouds for a given transformation.

(6) If error is smaller than the threshold decided before then terminate, else repeat step 2 to 5 (maximum of N times)

3.5 L_1-Medial Skeleton

Shape analysis and manipulation of 3D data become challenging in the presence of noise and missing geometry. One approach is to extract the skeleton representation from this noisy 3D data. We followed the same approach to use L_1-medial skeleton as 3D shape abstraction of the ice-hockey stick instead of directly using a noisy point cloud representation for the bending step.

L_1-medial skeleton [39] is a 1D structural representation for 3D point cloud data. L_1-medial has proven to be a great tool to get a global center of given data with an interesting property of being robust to outliers. This approach of L_1-medial skeleton makes an important observation to adapt L_1-medians locally for given input 3D points to generate a 1D structure. This calculated structure represents the localized center of the geometry or a medial curve skeleton.

If $Q = q_j$, where $j \in J$, be the given unorganized, unoriented dataset and $X = x_i$, where $i \in I$, be the optimal set of projected median points found using the L_1-medial skeleton definition:

$$\underset{x}{\operatorname{argmin}} \sum_{i\in I} \sum_{j\in J} \parallel x_i - q_j \parallel \theta(\parallel x_i - q_j \parallel) + R(X) \tag{29}$$

$$\theta(r) = e^{-r^2/(h/2)^2} \tag{30}$$

where the first term represents the localized adaptation of L_1-median of Q, R(X) is the regularization of the local point distribution of X. θ term in the expression that defines the weight function. This weight function is a fast decaying smooth function with the support radius of h which defines the local neighbourhood size for the skeleton generation. In the pipeline this approach helped us to minimize the effect of noisy point cloud on final template bending.

Chapter 4

Technical Contribution

As described earlier, capturing the bending in fast moving and structurally thin ice hockey stick is a challenging problem. Additional constraints of keeping the setup portable and low cost add to the complexity of the problem. We designed and developed hardware setup and software pipeline to address this problem [4]. This chapter will introduce the proposed hardware setup in Section 4.1, all components of proposed software pipeline in Section 4.2 and contribution of this thesis towards proposed pipeline in Section 4.3. Finally, in Section 4.4 we explain the

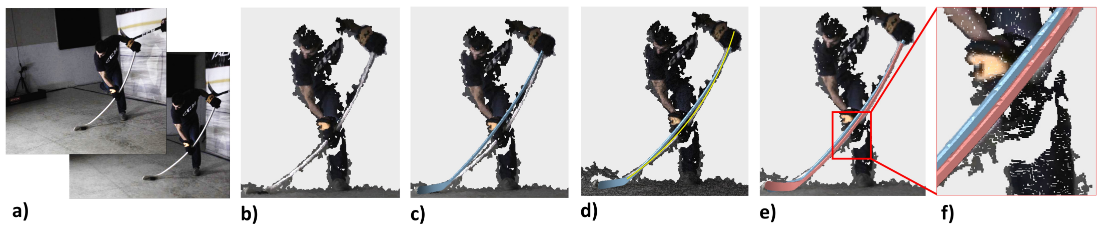

Figure 4.1: A visualization of complete pipeline. Figure-a) shows pair of stereo-images captured at an instance. Figure-b) showcases the reconstructed point cloud and Figure-c) shows the rigidly aligned stick template (blue) in the point cloud. Figure-d) illustrates the fitted quadratic curve in green. Figure-e) shows the bend template (red) and Figure-f) illustrate the zoomed portion for better visualization [4].

Figure 4.2: High speed cameras used for acquisition of the scene, Grasshopper 3.0 USB3 (courtesy:FLIR).

4.1 Hardware Setup

We designed a portable and low cost synchronised binocular vision based acquisition setup. This acquisition setup comprises of two Grasshopper3 USB3 color cameras from Flir and one custom made temporal synchronization box. These are vision cameras with global-shutter and temporal synchronization support. Ice-hockey is a motion intensive sport which is why we need high temporal acquisition to capture bending of the stick. With experimental testing on the floor, we came up with the 250 frames per second (fps) as sufficient temporal speed to capture majority of the bending. But we chose 275 fps as target frame rate as more experienced or skilled players can shoot at relatively higher speeds than normal.

To achieve this frame rate with relatively inexpensive cameras requires a trade-off between spatial and temporal resolution, which in turn significantly impacts reconstruction accuracy. In our setup in order to accomplish required frame rate, we chose a spatial resolution of 672×608.

4.2 Software Pipeline

In this section we introduce the complete software pipeline. There are different components involved in this pipeline which can be broadly classified as following, a) Point cloud reconstruction, b) Primary pruning (motion-based), c) Secondary pruning (proximity based), d) Template rigid-alignment, e) Medial axis generation, f) Final bending algorithm. Figure 4.4 showcases the logical

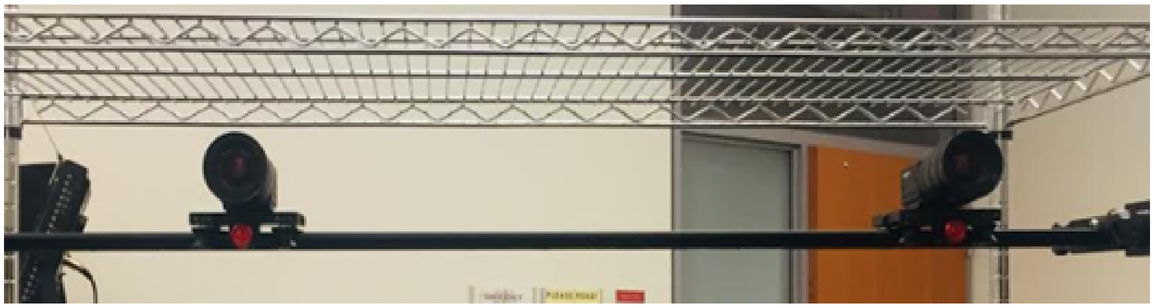

Figure 4.3: An illustration of the fast speed synchronized cameras setup mounted on the portable platform. The distance between the cameras can be changed as required. This system is customized to cater the need of easy portability.

flow of different steps in the pipeline.

For the first step, we used stereo reconstruction to obtain a point cloud per frame of the acquisition. However, this point cloud contains the stick and all the surrounding environment: the player, the rink, etc. depending on the setup. We perform two pruning techniques to get rid of all the unwanted points. In primary pruning, using the temporal information from the next frame, we prune the point that are static from frame to frame (Figure 4.5 right-middle). Then, we perform the rigid fitting of the template to the point cloud. As the stick undergoes a lot of deformation we cannot use the geometry of the straight stick as a template in all frames. Therefore, we use the reconstruction of the previous frame as the template for the current frame and we use the Iterative Closest Point (ICP) algorithm to rigidly align the template.

After the rigid template alignment, we perform the secondary pruning using a proximity based approach with this aligned template. Under secondary pruning, we prune the point cloud using the distance to the aligned stick template as a criterion (Figure 4.5 right-bottom). This is necessary because the pruning based solely on motion may still contain points belonging to some parts of the player body that are in motion (Figure 4.5 right-middle). We further need to deform the template stick to match the reconstructed point cloud. This deformation operation has two ingredients: a deformation model that can be applied to the stick geometry and a constraint mechanism that ensures the deformed template matches the point cloud.

For the physics-based deformation model, we use the co-rotational Finite Element Model (FEM)

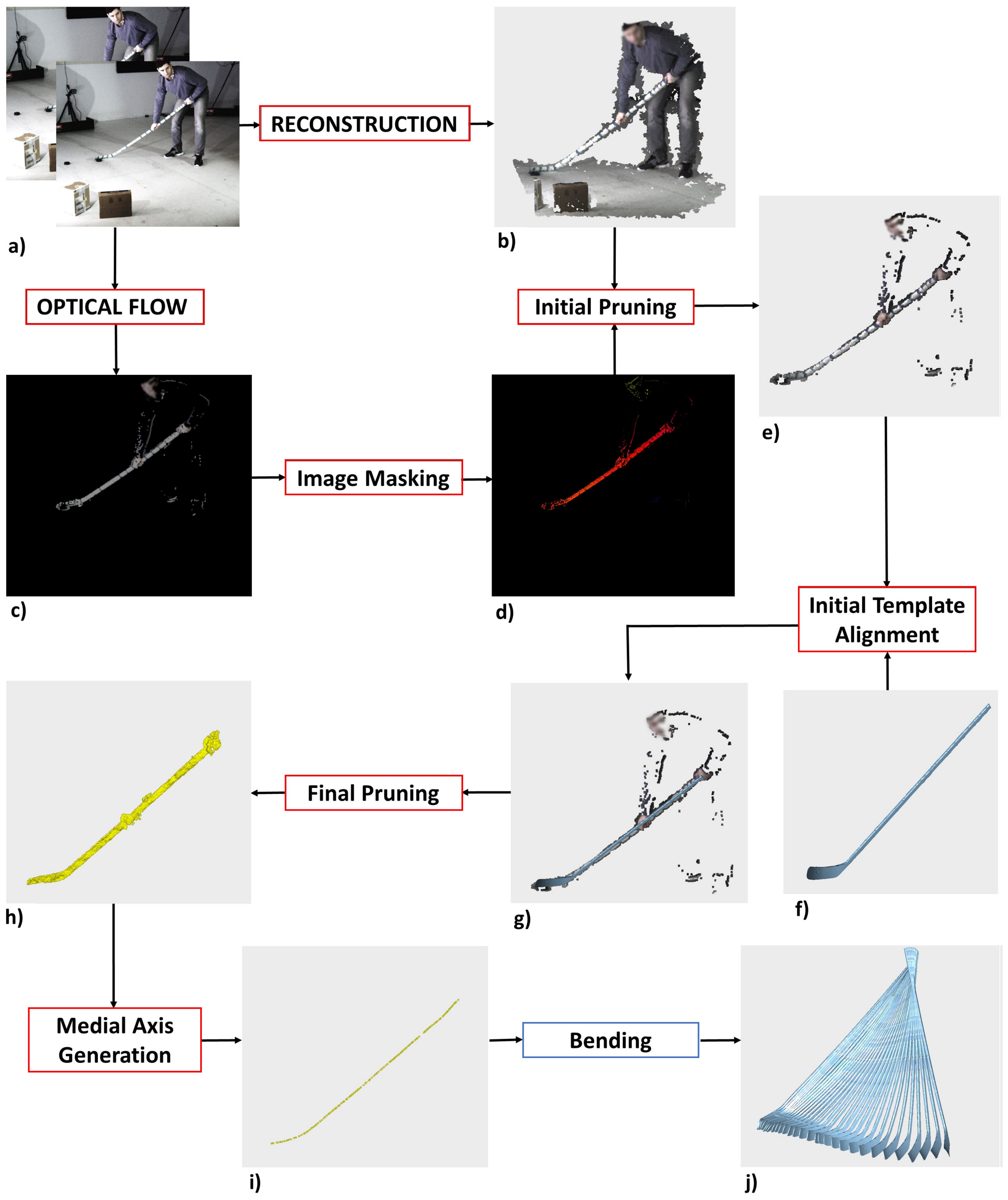

Figure 4.4: Illustration of the flow in the pipeline, 3D reconstruction, point cloud pruning, initial-rigid template alignment and media-axis creation (blocks in red) are the direct contribution of this thesis towards the overall pipeline. First using stereo images (a) reconstruction of point cloud (b) is performed. Initially pruned point cloud (e) is created by using image mask (d) from optical flow (c). Initial rigid alignment of stick template (g) is done using this initial pruned cloud (e) and provided stick template (f). After which the final pruning of point cloud (h) is performed. Medial axis (i) is generated for each pruned point cloud which guide the final bending in each aligned template stick (j).

method similar to [40]. While there is a rich set of deformation models, a FEM model was the most appropriate because we are aiming for a physically correct deformation. The co-rotational FEM model is accurate, efficient and numerically stable, particularly suited for deformation that is primarily rotational as it is in the case of the hockey stick.

The second part is to apply constraints to the hockey stick in order to deform it to fit the point cloud. However, the point cloud is very noisy due to (i) relative low resolution of the cameras, (ii) large distance from the subject and (iii) thin geometry of the hockey stick. Furthermore, the stick is partially occluded by the hands or gloves of the player. Therefore, using the point cloud directly results in artifacts as illustrated in Figure 4.26. To stabilize the geometry we robustly fit a quadratic curve to the medial axis of the geometry of the hockey stick and we use this curve to apply constraints to the hockey stick as illustrated in Fig. 4.1. Now we provide more details of our acquisition system and deformation process.

4.3 Contribution

This thesis contributes towards the ultimate goal of the pipeline in many different ways. First, designing a mobile stereo-vision setup and its calibration, capturing a lot of data acquisitions with different players shooting in different styles. Second, developing a two step pruning methodology to prune structurally thin and fast moving ice-hockey stick from noisy reconstructed point cloud in 3D. Third, automating the process of initial rigid alignment of the stick template in the noisy reconstruction. Forth, reducing the effect of noise by using medial axis approximation approach and suppressing the hand occlusion effect on the final template bending by a curve fitting approach.

4.4 Implementation

This section focuses on the fine details of the major components which are contribution of this thesis towards the proposed pipeline. Section 4.4.1 discusses the point cloud pruning methodology developed for this pipeline. Under this, we first discuss motion based pruning and then explore proximity based pruning. Then we move on to the details of the Global Initial-Rigid Alignment problem of stick template under Section 4.4.2. Finally, in Section 4.4.3, we discuss the need and

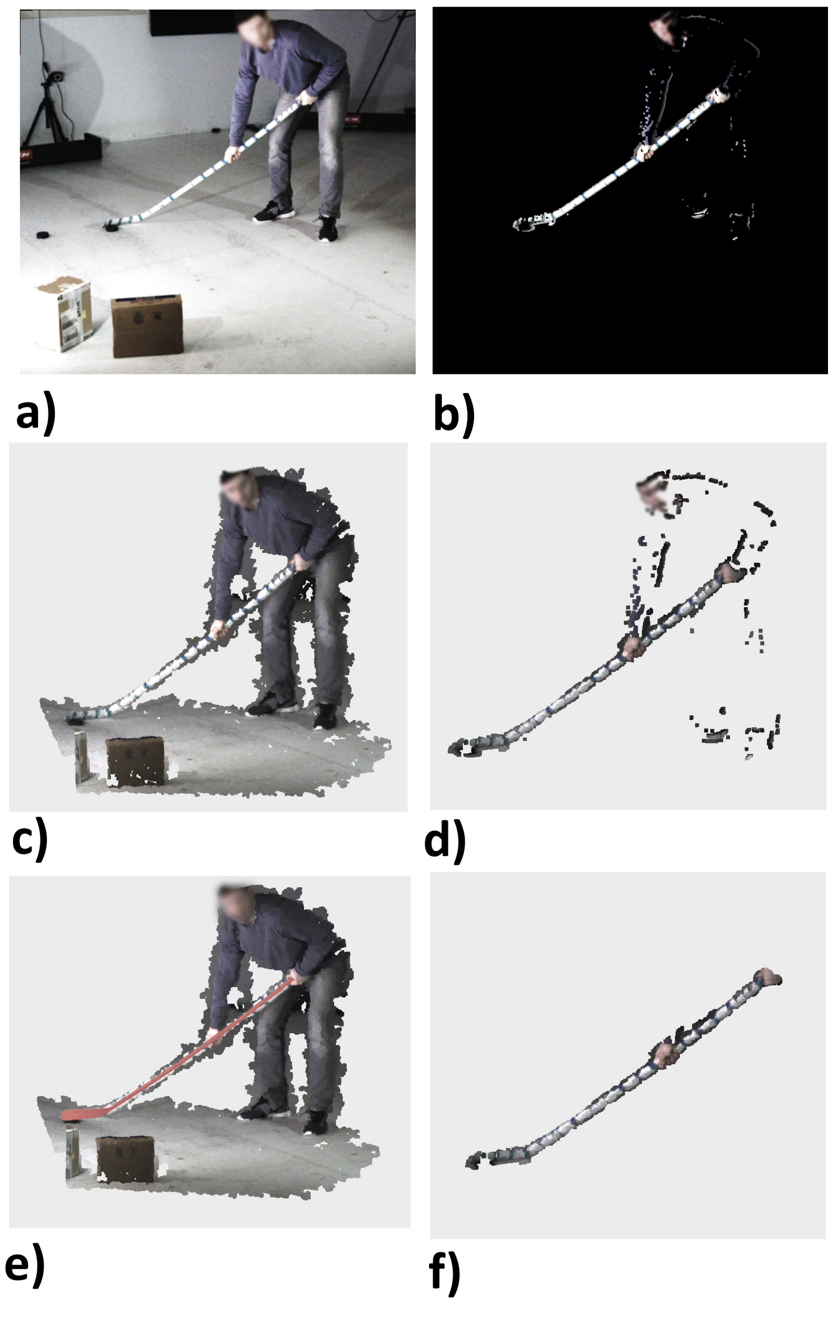

Figure 4.5: Complete Pruning Process. top-left) Input image; top-right) Masking out static pixels; mid-left) Initial point cloud; mid-right) Pruned point cloud based on the pixel mask; bottom-left) Initial point cloud with the aligned template; bottom-right) Pruned point cloud based on the proximity to the aligned template.

implementation details of Medial axis from the pruned point cloud.

4.4.1 Point Cloud Pruning

In Section 3.1 we have already discussed the point cloud reconstruction technique using binocular stereo-vision system. We generate point cloud for each frame of the acquisition using the same reconstruction technique. The point clouds generated contain points corresponding to ice-hockey stick (inliers) as well as the surrounding environment (outliers): player body, floor etc (can be seen in Figure 4.6 a). We have to prune (remove) all outliers because of the following reasons. First, it makes the pipeline lighter and reduces the per frame processing time. Second, to perform global registration of the stick template in the first frame automatically and third, to generate the required quality stick Medial axis for bending algorithm.

Ice-hockey is a very fast sport in terms of motion. When a player takes a shot with the ice-hockey stick, the bending in the stick happens for a very short time. To capture this fast bending motion of the ice-hockey stick, the setup requires a capture rate of at least 250 frames per second (fps). In order to process that frame rate with relatively inexpensive cameras, we trade off the resolution of acquisition images. With experimental testing we attained 250 fps with an image resolution of 672×608 for the existing setup. This spatial resolution trade off with other factors like, less availability of texture on objects and high space capture volume, significantly impacts the quality of point cloud reconstruction. Thus, making pruning of these point clouds with sacrificed quality a difficult task.

We successfully achieve the goal of pruning all the outliers from each point cloud using a two step pruning approach as depicted in Figure 4.6. Adding some extra constraints like specific color markers on the stick might reduce the complexity of the problem. But we try to make the pipeline as robust as possible without adding any extra constraints. We use the spatiotemporal information present in every acquisition sequence to segment the stick part only. Thus we settle for initial pruning of the point cloud based on motion using optical flow algorithm (see section 3.2). This initial motion based pruning removes a majority of static outliers from the point cloud, but still leaves some behind (see Figure 4.6b). Majorly, these outliers are moving points on the player's body or moving shadows found in the frames. Nevertheless, this initial pruning enables us to automatically

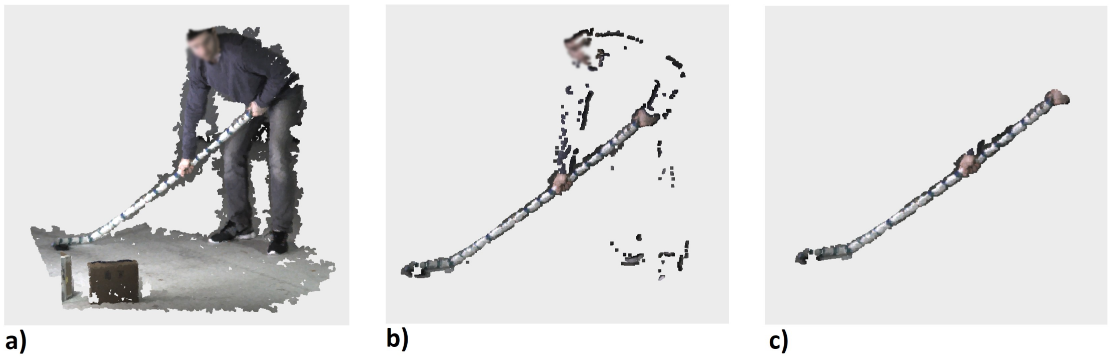

Figure 4.6: A visualization of sequential pruning of point clouds: Figure a) depicts the original point cloud. In Figure b), we see the primary pruning based on motion. There are still some outliers that are not removed. They are finally removed using secondary pruning based on proximity, as seen in Figure c). As a result of both pruning steps, the final pruned point cloud only retains points corresponding to ice-hockey stick.

position the stick template in the first frame (see Section 4.4.2) and paves the way for our final pruning approach. This secondary pruning is based on proximity with the aligned template (see Figure 4.6c). This two step pruning methodology remove all the outliers but pruned stick still have inbuilt noise from the reconstruction process. In next section we will see how to reduce this noise effect on bending by using the Medial axis approach (Section 4.4.3).

Pruning: Based on Motion

In the pipeline we capture the motion as sequence of frames. We use this spatiotemporal information to remove those outliers which are static in a pair of these sequential images. Let us say at any time, $T = t$ and $T = t + dt$ we acquire two time sequential image frames $I(t)$ and $I(t + dt)$ respectively. Here, time interval dt depends on the capturing frame rate. In our pipeline dt is very small (for frame rate = 250 fps, dt = 4ms). For such a small time interval dt we take an assumption based on our setup. We assume that motion captured between these two frames is majorly from stick movement with some noise (seen from Figure 4.7a). The sources of this noise are moving pixels that do not belong to the stick like player's body, object shadows, moving puck etc.

We will capture this motion using concept of Optical-Flow. Section 3.2 summarizes the theoretical background of Optical-Flow. For more robustness, we need to calculate the Optical-Flow at

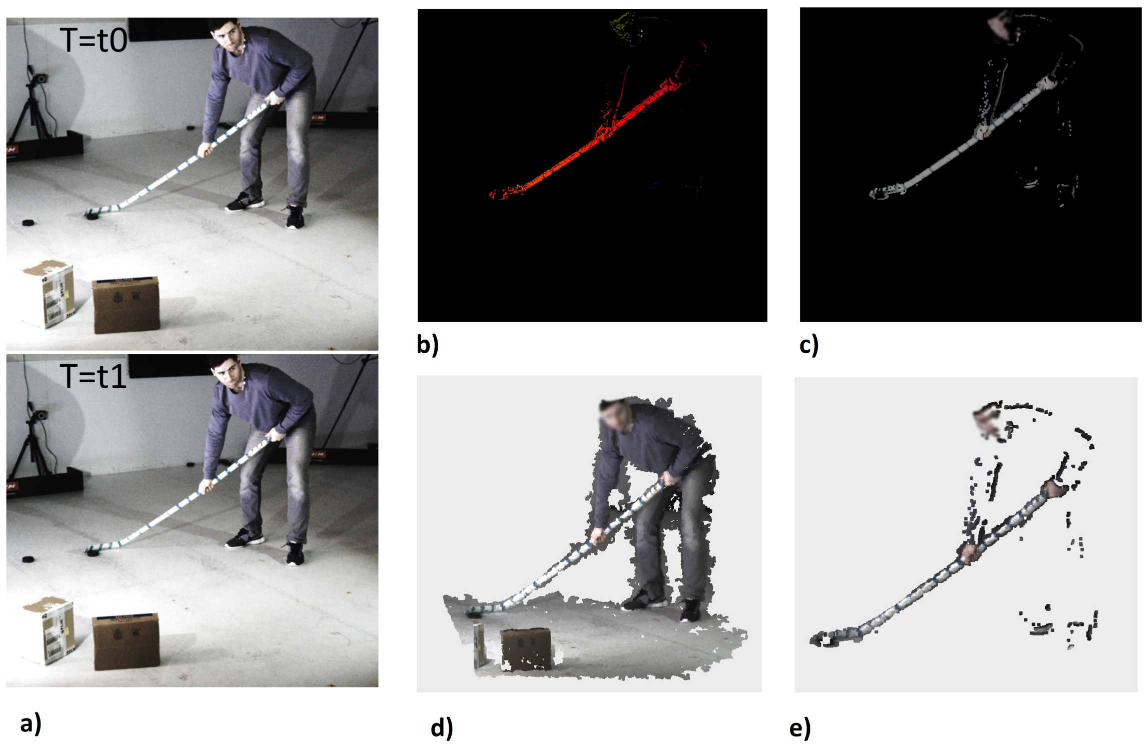

Figure 4.7: Explaining the motion based pruning technique of point clouds. Figure-a) shows two sequential images captured at time $t0$ and $t1$ from which we calculate the Optical-Flow (HSV representation) as shown in Figure-b). Figure-c) depicts the overlap of binary mask created with this Optical-Flow and original image. Figure-d) and e) compare the original point cloud, and the pruned point cloud respectively. Using motion based pruning technique a majority of static outliers are removed from the point cloud.

all pixel locations instead of some key-points. We have used Gunner Farneback's algorithm [36] for calculating Dense optical-flow using OpenCV [41] because of its efficient computation needed to process bulky data generated in each acquisition. In this two frame based motion estimation algorithm, the first step is to approximate neighborhoods of both frames by quadratic polynomials. We capture at very high frame rate therefore there is no big jump in pixels location in consecutive frames, in other words the motion captured within two consecutive frames is slow. We choose pixel neighborhood of size 8 to find polynomial expansion and an averaging window size of 15 as these values fit well to our slow motion estimation between frames. Since there is no fast motion in between two consecutive frames therefore we have used the pyramidal approach with just 2 levels with scaling factor of 0.5. On each pyramidal level algorithm iterates for exactly three times.

For each pair of immediate sequential frames of the acquisition, we have calculated Dense Optical-Flow in the form of two dimensional displacement vectors. For each pixel location I(x,y) we will get a displacement vector **V**=(V_x,V_y) depicting its motion. Figure 4.8a visualizes the Dense Optical-Flow calculated using a HSV (Hue, Saturation, Value) based color code where Hue value describes the direction.

Since, we are going to create an image mask for pruning the point cloud from the Optical-Flow, we have to further refine the Optical-Flow by doing some post processing steps. In this post-processing we use two approaches. First we filter noise from calculated Dense Optical-Flow by using a threshold value. Second, we use background subtractor algorithm to further refine it. To filter noise, we calculate the L2 norm for each vector as following

$$\left\|V\right\| = \sqrt{V_x^2 + V_y^2} \tag{31}$$

The speed of motion in the stick is slow at the starting and picks up in between before coming to a halt. Therefore, the value of resultant pixel motion in any two pairs of the sequence is not consistent. In order to settle down for a robust threshold value which can differentiate noise and moving pixel points for whole sequence we need normalization of values. Therefore, after finding the resultant for each vector we shift the values to same scale by normalizing them within the range of 0 and 1. We do a number of experimental trials, with data from different types of shots (Wrist and slap) and

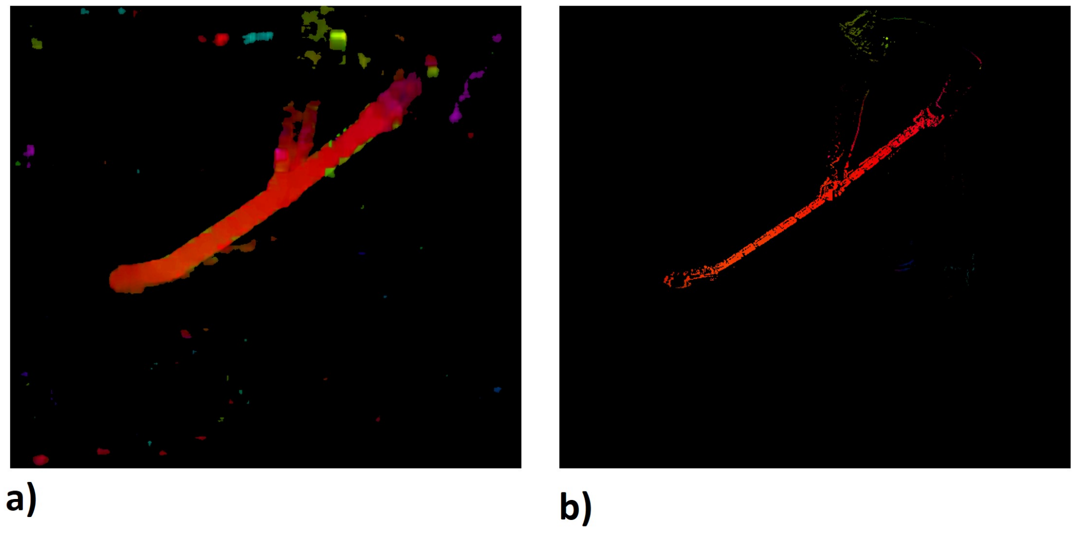

Figure 4.8: Comparison of original Dense Optical-Flow and its refinement after post-processing: Figure-a) shows original Dense Optical-Flow calculated using Gunner Farneback's algorithm while Figure-b) showcases the refined version after post-processing. Post processing steps involve threshold based noise filtering and background subtraction.

settle with the threshold value of 0.05. In order to create an Optical-Flow based binary image mask, we filter all the pixels with normalized value of the displacement vector of less than 0.05. Parallelly, we then create another binary image mask using a mixture of Gaussian based background subtractor using OpenCV. Lastly, a logical AND operation on both the masks gives us the final image mask (as seen in Figure 4.7b)

Now, we have an image mask containing moving stick and some outliers (which will be removed after the final pruning step). The next step is to prune the point cloud by applying the back projection technique. Section 3.1.2 touches on the theoretical background of the transformation matrix that we use in this technique. In a pin hole camera model, object with coordinates (X, Y, Z) is projected at location (u, v) on an image screen by using the following perspective transformation:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{32}$$

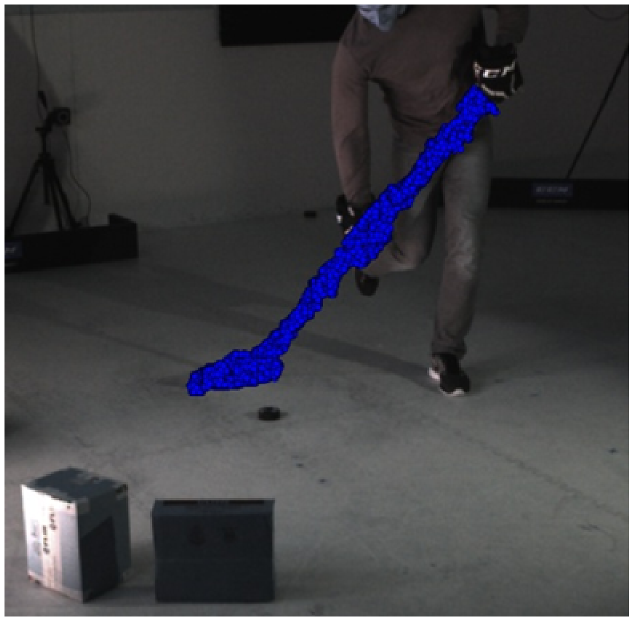

Figure 4.9: An illustration of back-projection of 3D point cloud on a 2D camera image. Blue dots represent the projected 3D points from pruned point cloud after final pruning step onto 2D image. For clarity of illustration, only 3D points corresponding to the stick are projected back on image.

In the pipeline we generate the camera transformation matrix after the camera calibration step in the Agisoft's Photoscan API. We use this information to back-project all (X, Y, Z) points of each point cloud to get the projected (u, v) pixel location of that point on camera screen.

After doing the back-projection, we have the set of (u, v) projected pixel locations of each point of point cloud and $(u^{'}, v^{'})$ non zero pixel locations from image masking. We then compare these two. We then perform a search operation between these two sets by taking advantage of kd-tree data-structure. Implementing this search operation with kd-tree greatly reduces the time complexity. We only keep those (X, Y, Z) whose projected pixel location (u, v) matches with non zero pixel locations $(u^{'}, v^{'})$ of masked images. In this way we have pruned the point cloud on the basis of the motion.

Pruning: Based on Proximity

In the previous section, we have explained the point cloud pruning technique based on motion. Pruning just based upon motion is not sufficient for this pipeline as outliers like players arm, moving

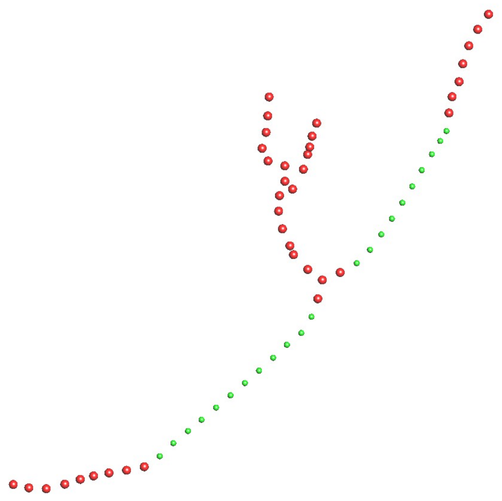

Figure 4.10: Showcases the outlier detection in the medial axis created just after the motion based pruning step that includes outliers like moving body parts of players. Red points are the outliers while green points are inliers detected using two step curve fitting approach. Because of this method's complexity and less robustness, a secondary pruning method based on proximity is preferred over this

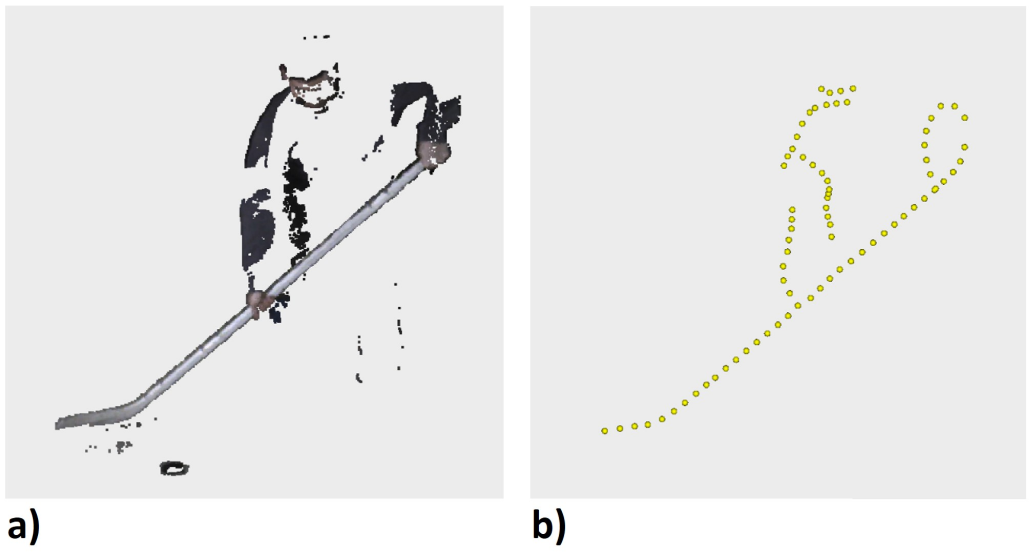

Figure 4.11: Need for proximity based pruning technique: Figure-a) shows pruned point cloud with primary pruning technique based on motion and Figure-b) depicts the Medial axis generated with the pruned point cloud. Medial axis generated with just motion based pruning technique contains a lot of outliers which further complicate the final bending algorithm.

puck are not pruned off using only motion based approach. But, this step is essential to locate the stick template in the first frame. Motion based pruning does not completely remove all outliers which leads to complex Medial axis structure. Figure 4.11 showcases the resultant pruned point cloud using motion based approach and the Medial axis generated from this pruned point cloud.

We can solve this problem using two approaches. Either remove these outliers before Medial axis generation or tackle them after Medial axis generation. We test both these approaches. Let's first talk about the later approach of removing outliers after generating medial axis. It's vividly clear that medial axis generated at this stage is more complex because of the presence of outliers (see Figure 4.11). We tried to remove these outliers from the generated medial axis using a two step curve fitting approach (as seen in Figure 4.12). Under this approach the goal was to first filter out the outliers be fitting linear curve and then filter the blade part by fitting a quadratic curve. This approach works well for most of the scenarios but this approach gets confused when a very complex geometry is generated by the presence of high number of outliers in certain motion pruned point cloud.

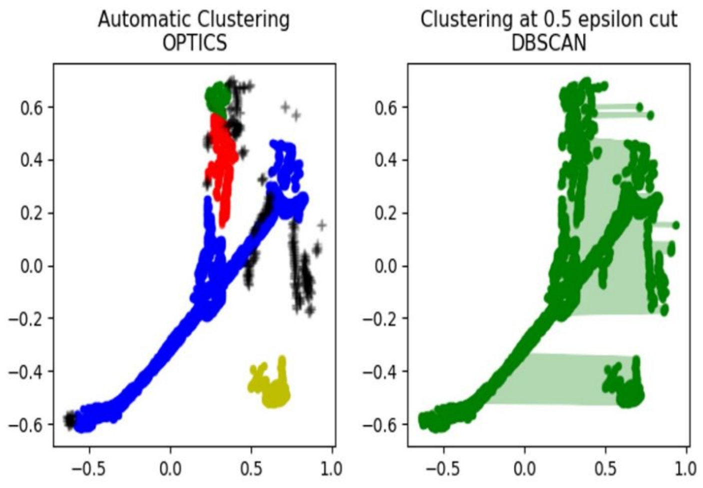

Figure 4.12: An attempt to prune using clustering based algorithm: Left figure shows the clustering based on OPTICS algorithm. Different colors corresponds to different clusters. Right Figure shows clustering based on DBSCAN with epsilon value of 0.5. It can be observed that these two clustering algorithms fails to correctly clusters inliers and outliers with desired quality

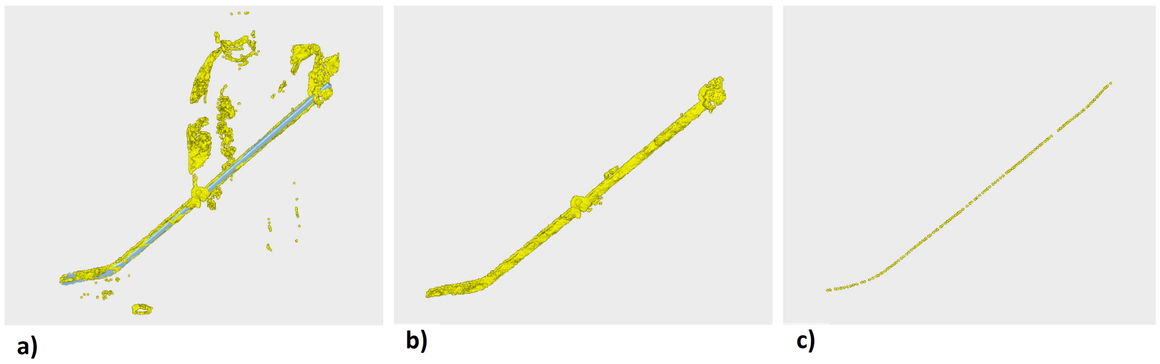

Figure 4.13: Secondary pruning of point cloud with proximity based technique. Figure-a) shows both the pruned point cloud (Yellow) obtained after primary pruning and the initially aligned stick template (Blue). Figure-b) showcases the point cloud after secondary pruning based on proximity with initially aligned stick template. Figure-c) illustrates the Medial axis generated with the final pruned point cloud. By using a secondary proximity based pruning technique all the outliers are removed from the point cloud and a clean Medial axis can be generated.

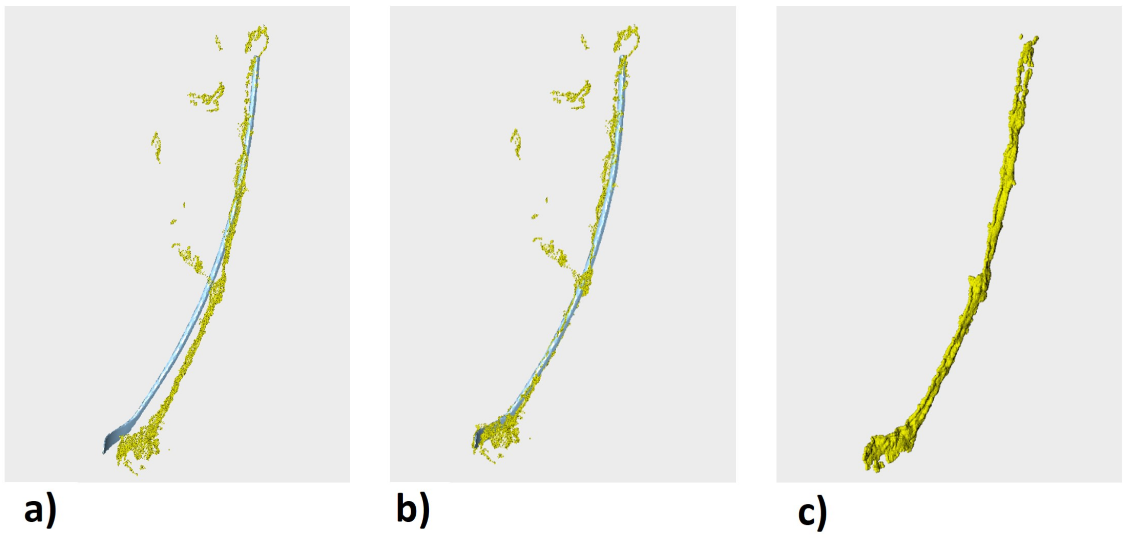

Figure 4.14: Adaptive alignment of stick template in proximity based pruning technique. Figure-a) shows primary pruned point cloud (Yellow) and stick-template (Blue) from previous frame. We then run the ICP algorithm on these two and obtain the adapted initial alignment for the current frame as shown in Figure-b). Figure-c) illustrates the final pruned point cloud generated with proximity based approach. We have now removed all the outliers which were present in the primary pruned point cloud.

Then we start trying with the approach of removing these outliers even before creating the medial axis. We tried a bunch of density based clustering approach like Ordering Points To Identify the Clustering Structure (OPTICS) and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (see Figure 4.12). But, it doesn't provide required quality results as not all the outliers are always distant from inliers. Then we inclined towards a lesser complex and more accurate approach. This time we utilize the information that we generated after locating the stick template in the first frame. In simple words, initial pruning (motion based) helps us locate the stick template in the first frame which in turn enables us to do the final pruning based on proximity to this initially aligned stick template.

After initial pruning with motion, we generate pruned point clouds with outliers for all frames. Using this initial pruning we locate the stick template in the first frame. For the first frame we use this initially aligned stick-template to prune. We construct a kd-tree with the points in the pruned point cloud. Then for each point vertex in the template we find all the points in the pruned point

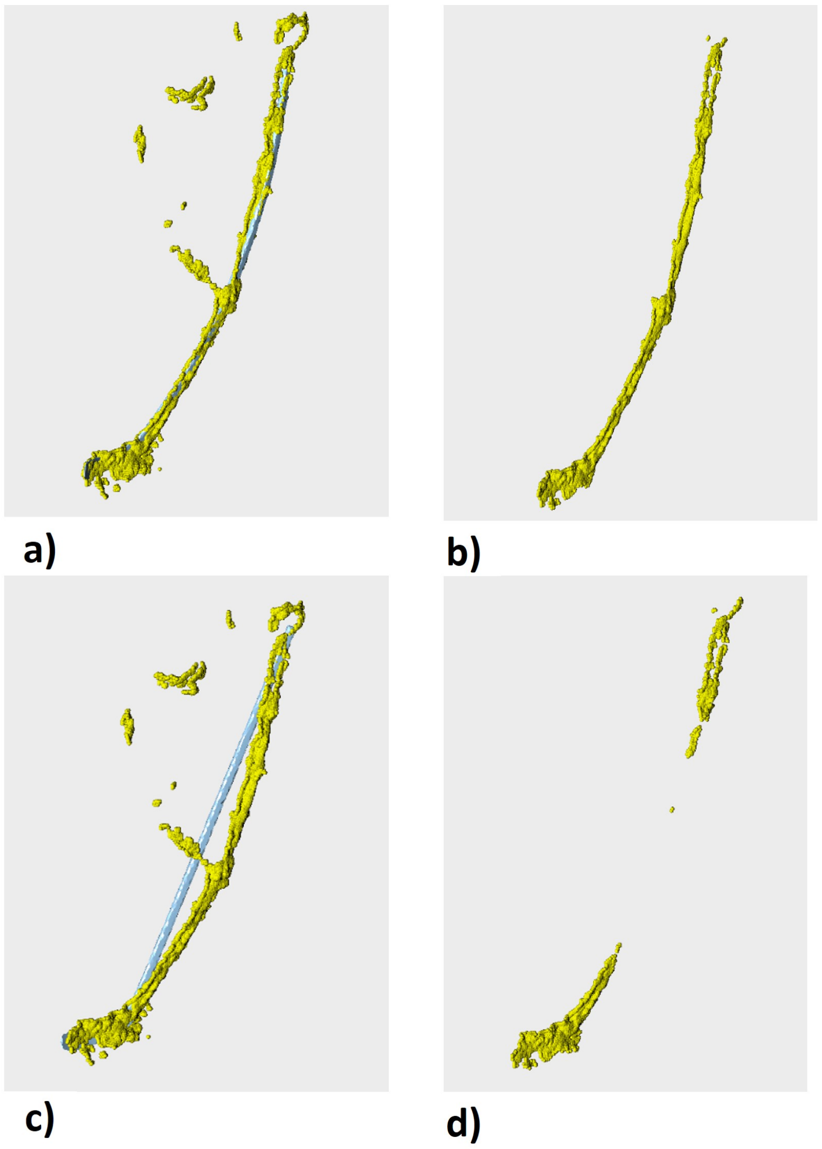

Figure 4.15: A comparison of results from proximity based pruning with and without the adaptive methodology. Figure-a) shows the point cloud (Yellow) with adaptive aligned bend stick template (Blue) and Figure-b) depicts the resultant pruned point cloud. On the other hand, Figure-c) showcases point cloud (Yellow) with aligned stick template (Blue), without any adaptive approach. Figure-d) shows final pruned point cloud without adaptive approach. It is clear that the non adaptive approach causes a loss of points from the bend region.

cloud, which lie within a certain radius. We keep only these points and delete the rest of them seen. With experimental testing we find that a radius of value 0.05 works well with our dataset.

In this pruning strategy we have implemented an adaptive methodology for all the frames apart from the first. For the first frame we get the template from initial alignment step. But for all the other frames, we use the bend template from the previous frame. The reason for using this adaptive methodology is demonstrated from Figure 4.15. The figure clearly shows a loss of points in the bending region if we use unbend template for each frame. To overcome this information loss problem we have implemented the adaptive method.

Under this adaptive method, we first run ICP (see Section 3.4) algorithm on bend template of the i^{th} frame and initially pruned point cloud of $i + 1^{\text{th}}$ frame. Thus, we get the initially aligned template for $i + 1^{\text{th}}$ frame. We then perform the same proximity based pruning method as explained for first frame. This way we get the new pruned point cloud for each frame which is free from outliers and suitable to generate the Medial axis.

4.4.2 Global Initial Rigid Alignment: Stick-Template

In this pipeline, we use a template based deformation technique to capture the bending in the ice-hockey stick. In order to perform the bending, this stick-template needs to be roughly rigidly aligned with the point cloud as shown in Figure 4.16. In this section, we explore how to achieve this initial rigid alignment of the template. We start with the initial alignment for first frame in Subsection 4.4.2 and then move on to propagate this aligned stick template for the rest of the sequence in Subsection 4.4.2.

Initial Rigid Template Alignment: First Frame

For the template based deformation technique, the stick template should be roughly aligned to the stick points of the point cloud. Since, the stick template is generated via different scanning methods and provided separately by the manufacturer, it can be present anywhere in the frame of reference of the point cloud. The only valid assumption that we can make is that the template is of the same shape and size as the one used in the acquisition. Figure 4.17 shows an example of the actual stick and its 3D-template mesh provided by the manufacturer.

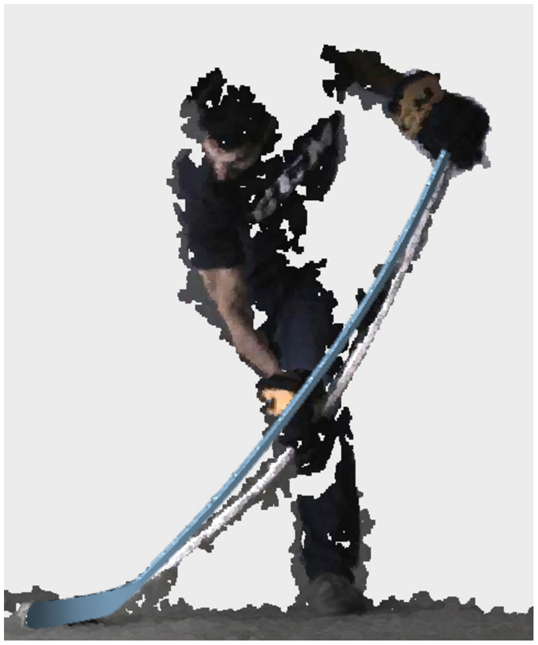

Figure 4.16: An illustration of rigid initial alignment of stick template. The Figure shows the point cloud with the initially aligned stick template in blue.

Let us frame this problem as a point cloud alignment problem. Consider we have a source point cloud (stick-template) and target point cloud (reconstruction) located somewhere in the global frame of reference. They can be any distance apart and in any relative rotation with respect to each other. Here, the objective is to find a translation and rotation matrix that would roughly rigidly align them together.

But there are a couple of challenges here. First, the target point cloud (reconstruction) consists of many outliers in addition to the source object: floor, player body etc. Second, the source and target point clouds differ significantly in terms of quality as both are generated by different scanning techniques. Source (template) point cloud provided by manufacturer is well built with no missing parts, while the target (reconstructed) point cloud is of sacrificed quality because of the reasons explained before. Figure 4.18 explains the significant difference between the source and target point cloud in terms of quality.

In order to remove the outliers from the target point cloud we implemented an initial pruning step based on motion (see Section **??**). After this pruning step, we removed most of these outliers. Now, we use a technique of feature matching to solve our initial rigid template alignment problem. There

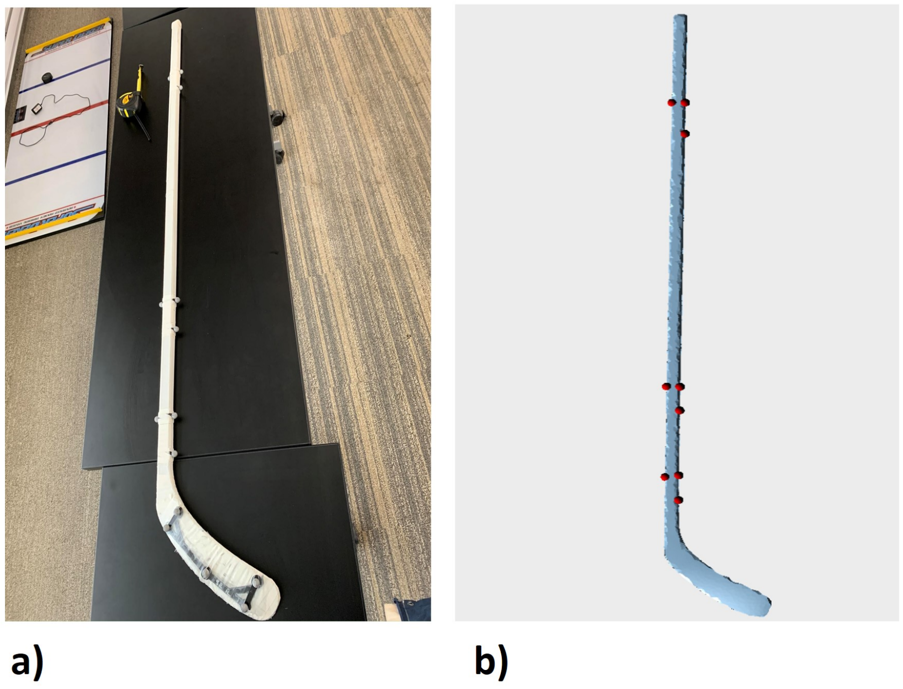

Figure 4.17: Stick template: Figure-a) shows actual ice-hockey stick used in an acquisition and Figure-b) showcases 3D mesh based template of the actual ice-hockey stick. Different sticks will have different stick templates.

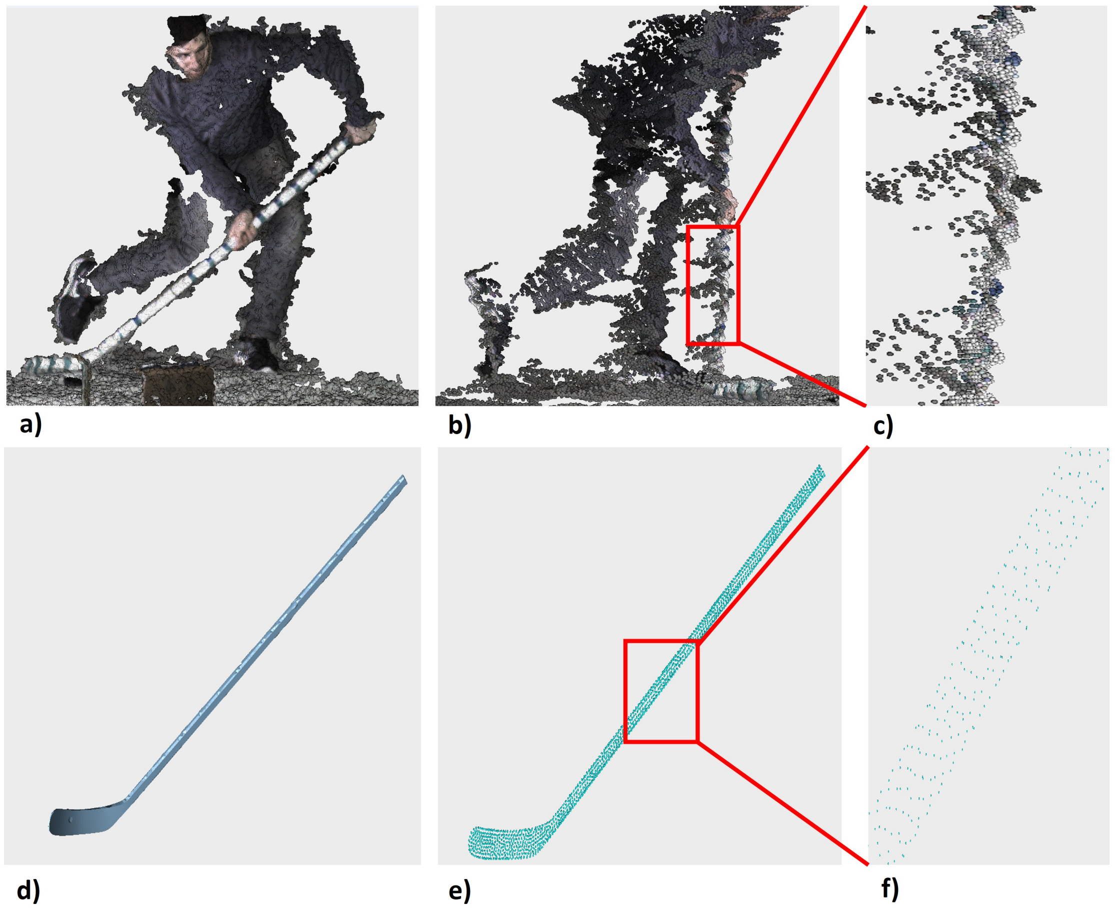

Figure 4.18: A comparison of mesh based stick template with the reconstructed point cloud. Figure-a) shows a reconstructed point cloud while Figure-b) shows the same reconstructed point cloud in a different view. A zoomed portion of Figure-b) can be seen in Figure-c). Figure-d) visualizes a mesh based stick template and Figure-e) showcases its Vertex distribution. Figure-f) is the zoomed representation of Figure-e). As vividly seen from Figure-c) and f), stick template point distribution is more uniform while reconstructed point cloud is noisy and irregular. This irregularity complexifies the point feature mapping.

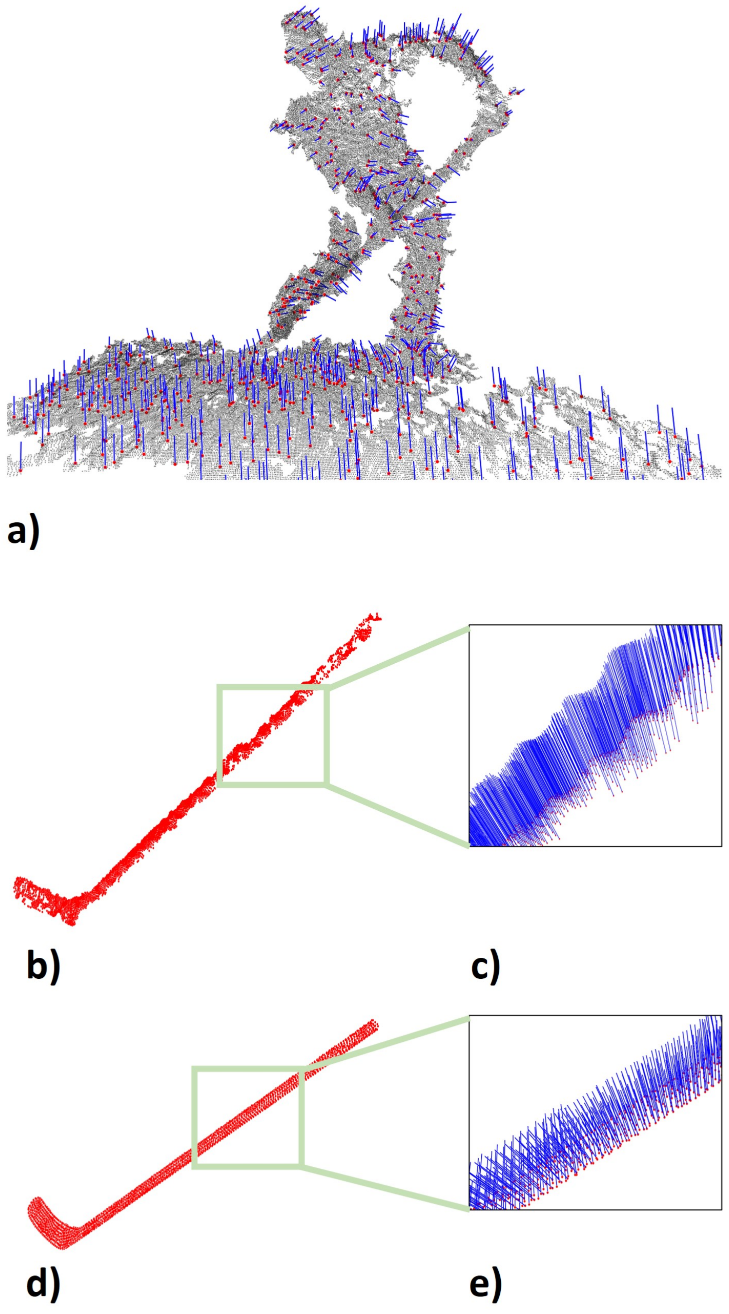

Figure 4.19: Point Normal Visualization: Figure-a) shows point normals (blue lines) of a down-sampled point cloud (red points). Points in black show original point cloud before downsampling. Figure-b) visualizes point cloud vertices (red) after final pruning while Figure-c) showcases a zoomed portion with point normal distribution (blue lines). Figure-d) shows vertex distribution (red) of stick template mesh while Figure-f) showcases the zoomed-in portion with normal distribution (blue lines). Pruning facilitates for initial stick alignment as with pruning we remove the peripheral objects from point cloud (like floor, players leg etc) whose normal distribution might falsely match rather than with the actual stick points' normal distribution during point feature matching.

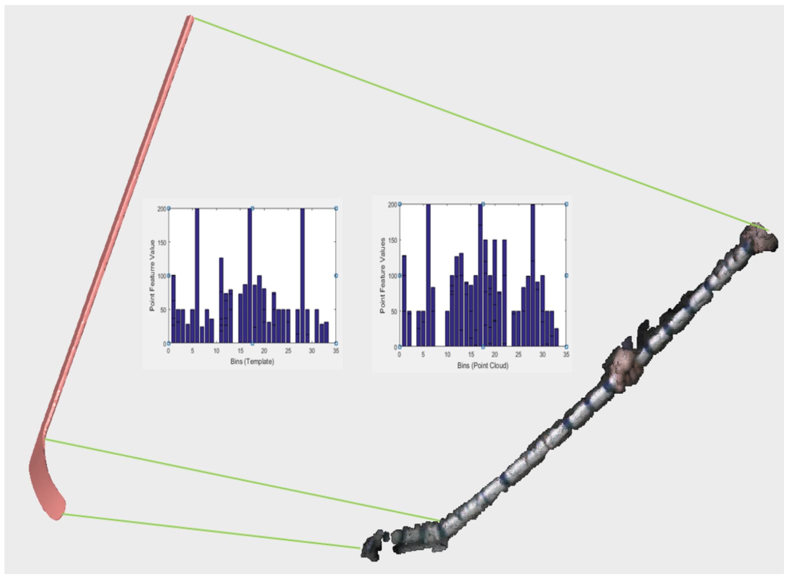

Figure 4.20: A visualization of Point Feature Histogram. Figure shows stick template (Left Stick) and its point feature histogram for a random point (Left chart), while the figure shows final pruned point cloud (Right Stick) and its point feature histogram fora random point (right chart).

exists a large number of 3D point feature descriptors like, Signature of Histogram of Orientation (SHOT), SIFT-3D, Spectral Histogram and FPFH (see Section 3.1.3). We choose FPFH features as these point features are pose and point cloud density invariant, computationally less expensive and more robust to noise. We use Open3D [42] library implementation of FPFH and RANSAC (see 3.3) to solve it.

Finding point correspondence: Fast Point Feature Histogram (FPFH)

This initial alignment problem becomes easily solvable if we know the perfect point-to-point correspondences in source and target data sets. However, this is not the case here as we are not using any artificial markers. We have to robustly estimate the good correspondences between the source and the target. We achieve this by using FPFH. Section 3.1.3 briefly describes the basics behind creation of this multi-dimensional features.

Before calculating the FPFH we have to do some pre-processing. Firstly, we downsample both the source and the target point cloud using a voxel based downsampling scheme. This downsampling helps us to save both time and space. By doing the experimental trials on different samples of our dataset, we settle on a voxel size of 0.05. After downsampling, next step is to calculate the surface-normals for both the source and the target. We find normals by best fitting the plane to the neighbourhood surface using principal component analysis. We use a radius of $5 \times voxelsize$ to search for neighbours. Figure-4.19 visualizes the normals in the source and the target point clouds. After generating normals, the next step is to find the Fast point feature histogram for all the points in both the downsampled source and target point clouds. We construct a kd-tree based data structure for the target point cloud feature map. In the next section we explain how we use this data structure for faster searching operations. Figure-4.20 shows an example of the Fast point feature histogram created as described above.

Finding Rigid Transformation: RANSAC approach

To find the final rigid transformation (translation and rotation) matrix for the source point cloud so as to align with target point cloud, we use iterative approach of Random Sample Consensus (RANSAC). A general introduction to the RANSAC is provided in Section 3.3. In the last section, we explained how to generate FPFH signatures for each point in the source and the target point cloud. We now find the transformation matrix using these point features iteratively.

For each iteration, we randomly find 4 points in the source point cloud. The only condition on these points is that they have to be at-least $3.5\times$ (voxel size) distance apart. Then, for each of these points we find a list of points in the target point cloud whose FPFH is similar. From these, we randomly select one point, which will become the source's point correspondence. After finding the correspondences of all these points we construct the rigid transformation and compute an error metric for the whole point cloud. This error metric computes the quality of the transformation. We use an RMSE (Root mean square error) based error metric. After rigid transformation, for each point in the transformed source we find the closet neighbour point in the target point cloud. We then calculate the RMSE using the Euclidean distance for all points and save the transformation matrix with the error value. We iterate the whole process 40,000 times and find the best fit transformation based on these error values.

Initial Rigid Template Alignment: Remaining Frames

In the previous section, we explained the idea of initial rigid alignment of the stick template in the first frame. We now discuss how to propagate this initial alignment in all the other frames of the sequence. For doing so we use Iterative Closest Point (ICP) algorithm. Section 3.4 describes the theoretical background for ICP algorithm. After the initial rigid alignment we have a stick template aligned to the first point cloud of the sequence. First, we run the ICP on the same point cloud for fine tuning the alignment. We then run the bending algorithm to get the bend stick template. We then again run ICP on this bend stick and the next point cloud in the sequence. In simpler words, we run the ICP on the bend template from the previous frame with the point cloud of the current frame. Thus, by following this adaptive procedure for all the frames we propagate the initial alignment of stick template to the complete sequence. Figure-4.21 shows the final aligned templates for the complete sequence.

4.4.3 Medial Axis Generation

Shape analysis and manipulation is a very complex process. In this pipeline, we have to analyze the shape of the deformed ice-hockey stick using point cloud reconstruction and model this deformation using a mesh template (ice-hockey stick template). But, we have to sacrifice on the point cloud quality because of frame rate and less texture. This means that the point cloud that we generate are noisy and have some build-in artifacts. Figure 4.22 shows the various artifacts in the input point clouds. If we use these noisy point clouds as it is in the pipeline than we would get non-realistic results in bending as can be seen from Figure 4.26.

In order to solve this problem, we have to extract the exact bending shape from noisy point clouds. The technique we use to solve this problem is Medial axis skeleton approach. Extracting a skeleton representation from a 3D shape is known to be an effective means for shape abstraction.L_1-median is known to be a vital statistical quantity that extends the univariate median to multivariate environment. It represents a global unique center of given dataset and known for its robustness to outliers and noise [39]. Adapting L_1-medians locally in a dataset representing a geometrical structure results into a one-dimensional structure called as L_1-medial skeleton [39]. Since in this

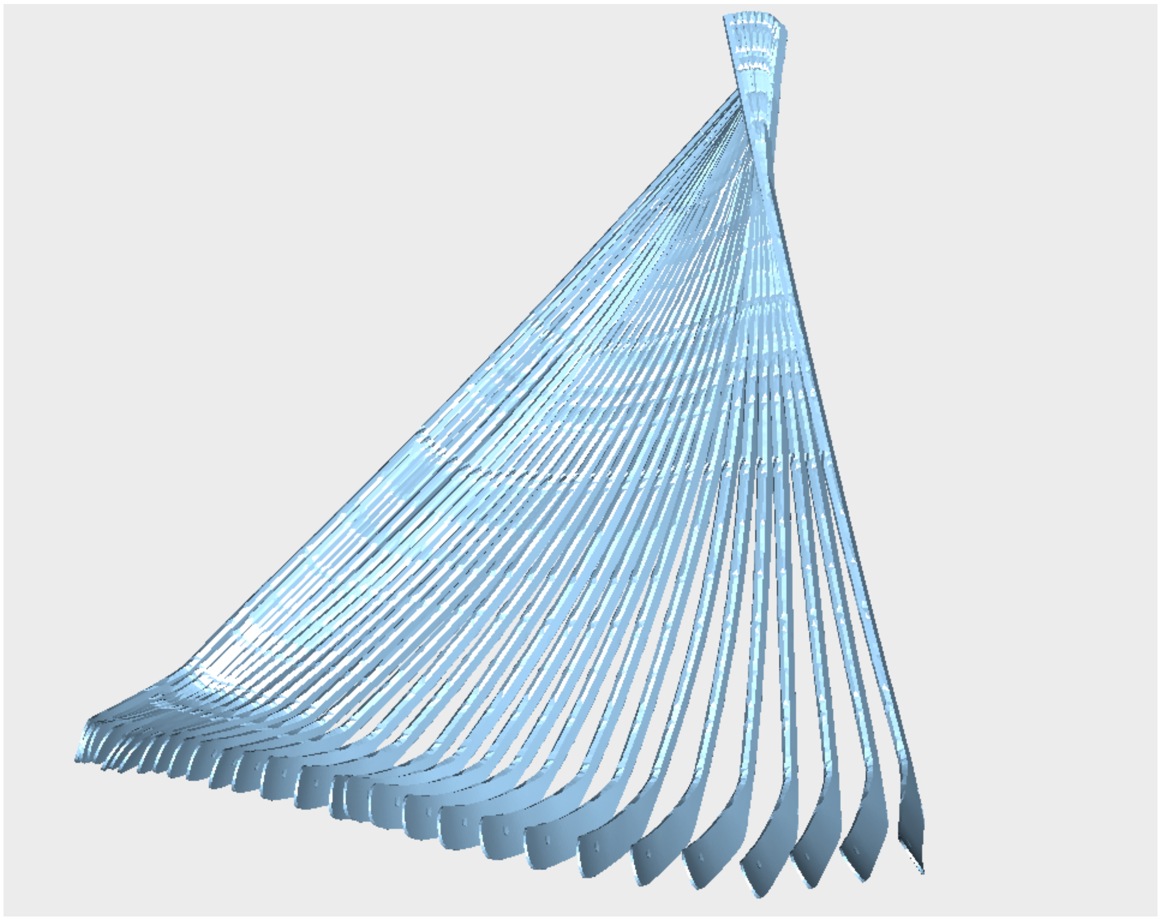

Figure 4.21: Propagation of initial alignment of stick template from first frame to the entire sequence, using ICP with the adaptive approach.

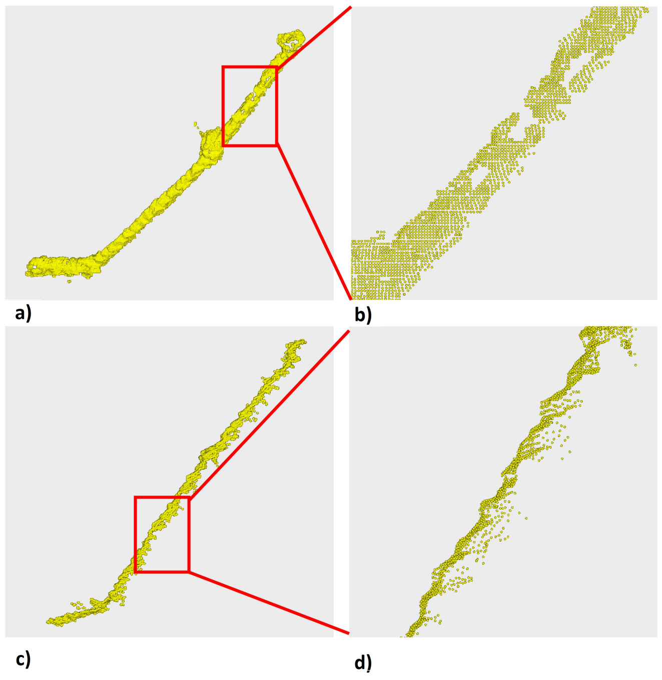

Figure 4.22: An illustration of artifacts in point cloud reconstruction. Figure-a) shows a pruned point cloud and Figure-b) showcases a zoomed portion of the same point cloud highlighting the missing portions. Figure-c) shows the same pruned point cloud in a different view and Figure-d) showcases the zoomed portion highlighting non-uniformity in reconstruction.

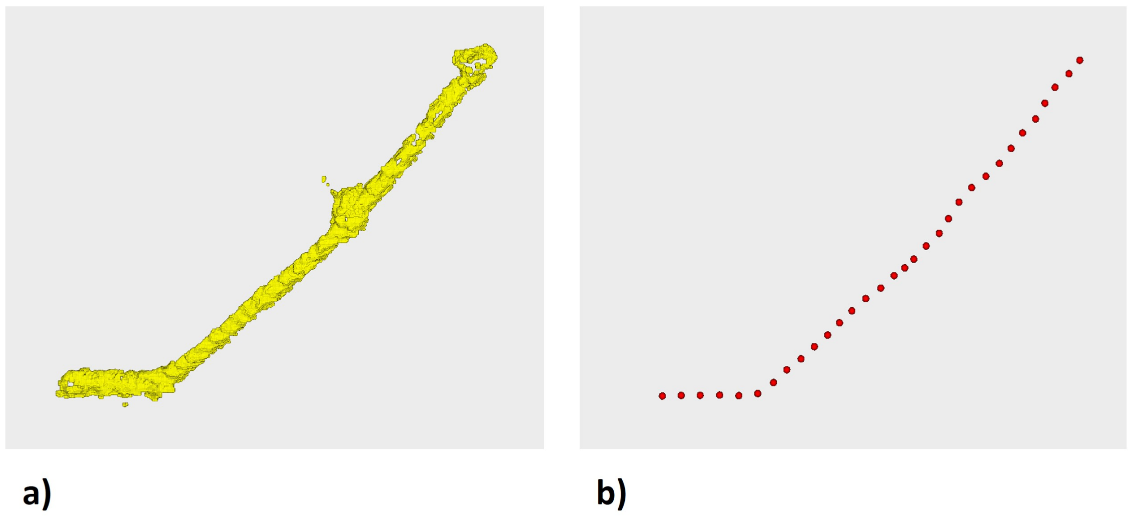

Figure 4.23: Medial Axis Generation. Figure-a) shows a pruned point cloud and Figure-b) showcases the generated Medial axis skeleton curve from the pruned point cloud. We reduced the dimensionality of the problem to abstract the shape from the object.

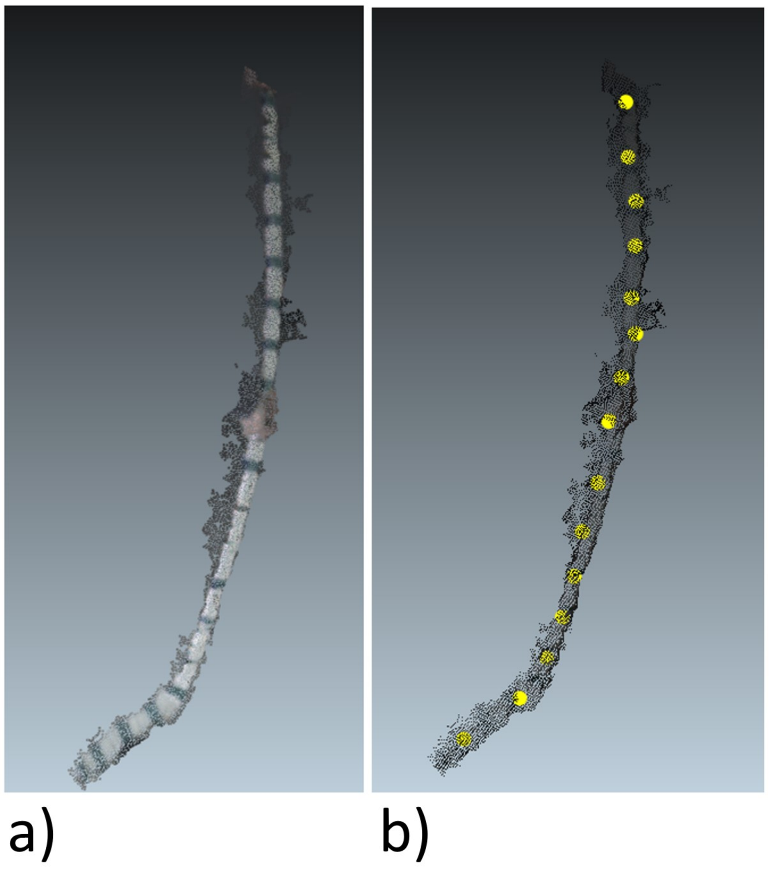

Figure 4.24: Noise in medial axis. Figure-a) shows a pruned point cloud and Figure-b) showcases the generated Medial axis skeleton curve (yellow) overlaid with the pruned point cloud. We can observe the little bump shaped distortion at the center because of hand occlusion and non uniformity in medial axis because of presence of small outliers in the pruned cloud

pipeline we are dealing with noisy point cloud with inconsistent point density and missing geometry which makes L_1-medial skeleton a good fit to reduce the dimensionality of the structure. Section 3.5 explains in detail the actual algorithm used to generate this Medial axis skeleton curve.

After we perform final pruning step, we obtain a nicely pruned point cloud that contains just the stick part. We create a Medial axis skeleton curve from this pruned point cloud. In order to generate this Medial axis, we first have to downsample the actual point cloud. After downsampling the point cloud, we use the implementation provided by [39] to generate the Medial axis because it operates on the raw data scans, provide fast and robust skelton extraction. Figure-4.23 shows the input pruned point cloud and the Medial axis generated. By using medial axis approach we suppress this noise by reducing the dimensionality.

But, still there exists a couple of challenges that push us back from getting smooth bending results directly from this medial axis. First, is the occlusion by hand or hand glove of the player. If we closely analyze the Figure 4.24, we will see that there is little outward bump in the middle of medial axis. Reason for this distortion mainly account from the continuous occlusion from the player's hand. Second challenge is the non- uniformity in the medial axis. As with medial axis creation a significant amount of noise is compressed but still the level of noise present in medial axis is significant enough to disturb smoothness of the bend.

Many attempts were made to smoothen the median curve and one of the failed attempt was to use a regression spline to fit the medial axis curve but fails as it try to fit the same curve both in blade and shaft points which results into bad curve fitting. A simple but effective solution to this problem is fitting a smooth curve to the medial axis. We use a quadratic curve as bend in the stick is most accurately represented by it.

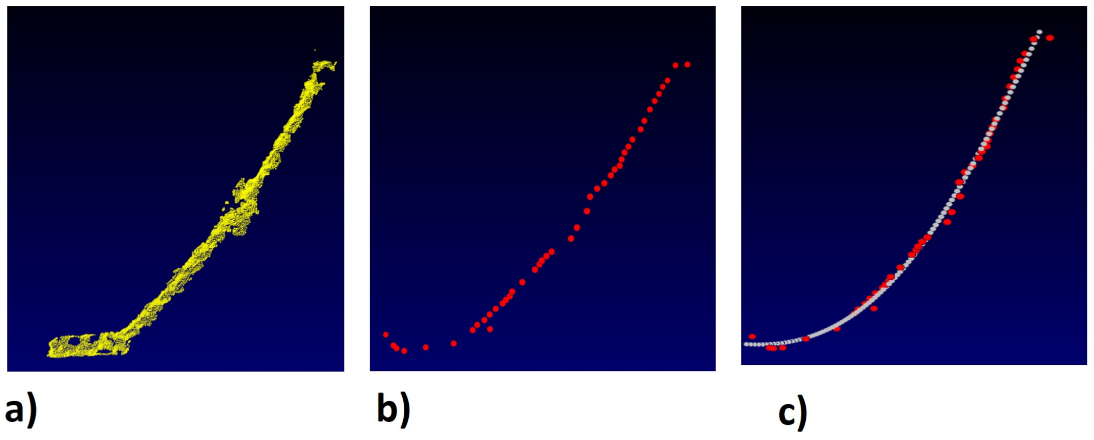

Figure 4.25: A failed attempt to smoothen medial axis by fitting a regression spline. a) shows a pruned point cloud of ice-hockey stick, b) represents a medial axis of the pruned point cloud and c) showcase the fitting of a regression spline (gray) to the medial axis. A close view at the bottom part shows that the curve twist a bit to accommodate the blade part which is geometrically wrong for the bending.

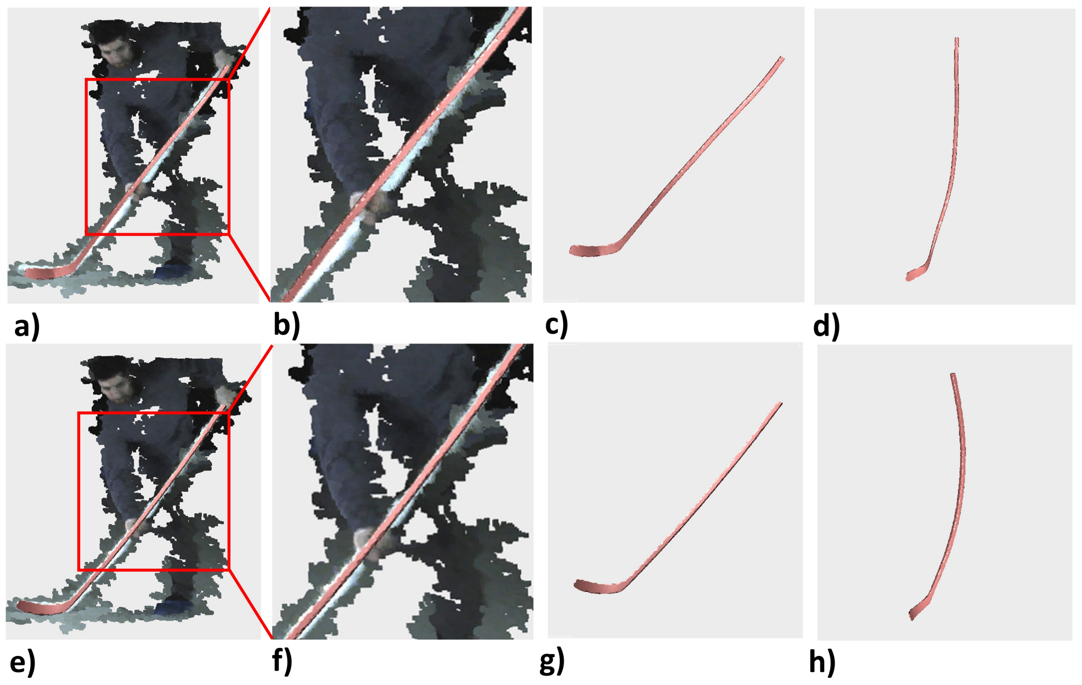

Figure 4.26: Comparison between fitting the point cloud directly or using the medial axis on a given frame. a-d)Result fitting the point cloud directly. e-h)Result using the medial axis. The results when using the point cloud directly exhibit clear undesirable artifacts while the result using the medial axis conforms well to the real stick shape.

Chapter 5

Results

In this chapter we provide qualitative and quantitative results on the bending of the stick. Figure 5.1 and 5.2 showcases the qualitative evaluation by overlapping the reconstructed scene point-cloud with the resulting mesh. As observed from a visual inspection, the deformed stick template closely matches the point-cloud.

The current state-of-art method to quantify the bending of the stick is with MOtion CApture (MOCAP). In this method a number of markers are attached to the hockey stick and a sate of art MOCAP is used to determine the various parameters of the stick like the maximun bending of the stick. Although the MOCAP systems are very mature and accurate, they do not capture the entire stick, only the trajectories of a few points. Furthermore, since our cameras are color cameras and the MOCAP system's cameras are infrared cameras, it is very difficult to align spatially the coordinate frame of the MOCAP system with very high accuracy. Lastly, it is very challenging to temporally synchronize the MOCAP system's data stream and our data stream.

However, the maximum bend of a stick, an important benchmark for the stick deformation is independent of spatial and temporal alignment so we decided to focus on this for our quantitative evaluation. We follow the standard industry method used to estimate the maximum bending angle. As shown in Figure 4.17 we attached a number of markers on the shaft in a pattern that creates three local coordinate frames A to C. Two axes are formed from the markers and the third one can be obtained using a cross product operator. The frame is then normalized to obtain an orthonormal matrix. We computed two angles: angle 1 between the coordinate frames A and B and angle 2

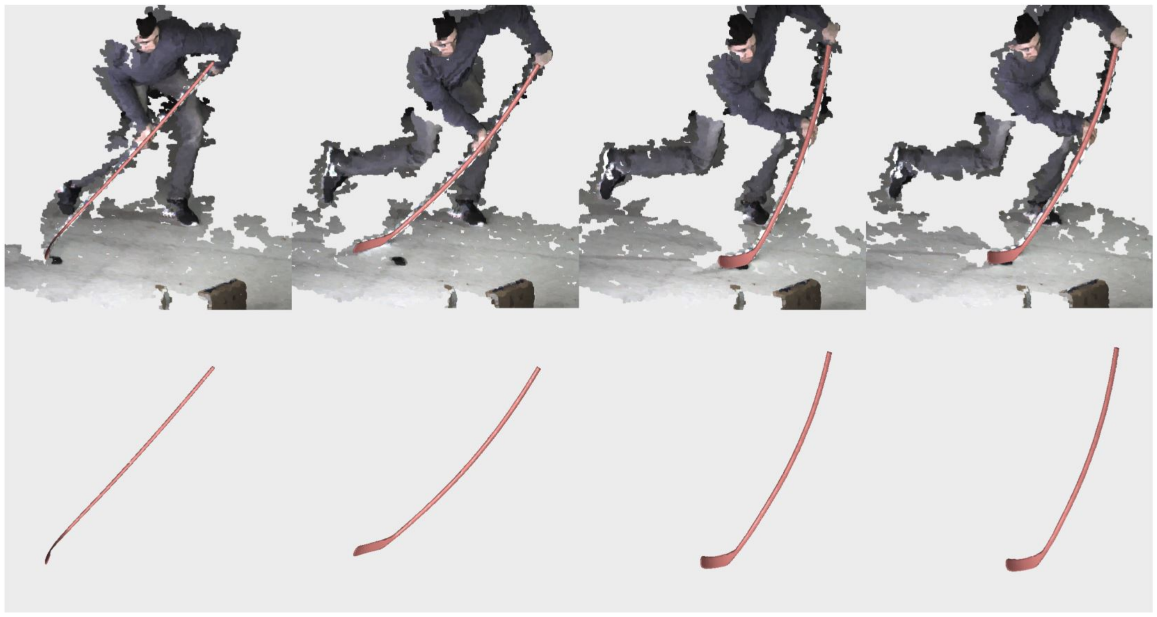

Figure 5.1: A few frames from a wrist-shot. We can capture the stick orientation both in deformed and non-deformed configuration.

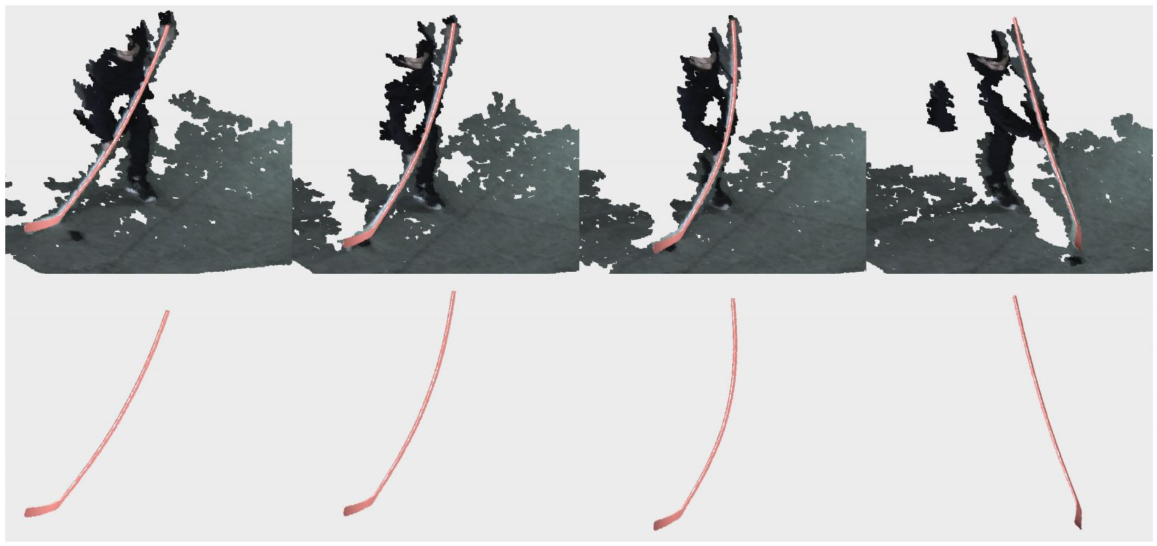

Figure 5.2: A few frames from a slap-shot. We can even capture the backward bend in the last frame due to the stick oscillations in the high energy shot.

	Slap Shot	**Wrist Shot**
Max Angle MOCAP 1	21.8	20.0
Max Angle Our System 1	23.65	22.05
Diff 1	1.85	2.05
Max Angle MOCAP 2	26.2	23.15
Max Angle Our System 2	28.35	26.65
Diff 2	2.15	3.5

Figure 5.3: A comparison of our bending results with a MOCAP system.

between coordinate frames A and C. We mimicked the same setup on the virtual stick by manually positioning marker points. Since the the MOCAP markers are physically outside the stick, this is a source of errors. Nevertheless, we averaged the two maximum angles for 2 slapshots and 2 wrist-shots using both systems. The results are presented in 5.3. The expected difference in the maximum bending angle is about 4 degrees. It is important to note that some of this error arises due to somewhat imperfect alignment of real and virtual markers.

The running time is dominated by the bending step that takes on average 87 seconds per frame. All the rest of the steps combined take less than 15 seconds per frame for a total combined of about 100 seconds per frame.

Chapter 6

Conclusion

Capturing the bending in the ice-hockey stick is one of the crucial parameter for the stick's performance analysis. There are many big challenges on the way to accurately reconstruct the bend created by a player in the ice-hockey stick while playing an actual shot. In this thesis, we present an idea of a complete software pipeline with hardware setup for reconstructing the 3D deformations in the template shape of an actual ice-hockey shot. We start by capturing a passive-stereo based acquisition of a complete ice-hockey shot sequence. We then process it by first reconstructing the complete scene using stereo-vision technique. Then, segment just the ice-hockey stick in 3D using a two step pruning methodology. After pruning, a medial axis of pruned stick is generated for each frame and a quadratic curve is fitted to recover the smooth bend from the noisy point cloud reconstruction. Finally a stick template which is automatically initially rigid-aligned in reconstructed scene is deformed using a co-rotational FEM model to match the bend curve derived before.

This thesis contribute significantly towards the ultimate goal of the complete pipeline. We did a lot of data acquisition of different shots with different players to enhance the robustness of the system. We developed, iterated and compiled the different sub-parts of the software to a single automatic pipeline. This thesis explains in great detail the exact pruning methodology adopted to prune a thin structure object from a noisy point cloud reconstruction. It also describe the process to initial rigid alignment of the stick template in a noisy point cloud reconstruction using the technique of FPFH 3D feature matching with RANSAC approach. Further, it also suggest an innovative method using the medial axis to remove the noisy point outliers and accurately recover the bend shape from the noisy ice-hockey stick reconstruction. We tested this pipeline with different stick objects, shooting styles and it perform well in all cases.

6.1 Future work

In this research work we focus solely on the bending of the shaft part of the stick and not on the deformations of the blade part. A future work would include segmentation and deformation of the blade part of the stick which is quite challenging because of blade's geometrical proximity with puck, floor and partial occlusion by the puck. The other limitation of this pipeline is the over dependence on the initial reconstruction of the point cloud which is too noisy. In a future work, one can replace the stereo reconstruction with a visual hull reconstruction using image-based stick detection in each image.

Bibliography

[1] R. Jain, R. Kasturi, B. Schunck, and B. Schunck, *Machine Vision*. Computer Science Series, McGraw-Hill, 1995.

[2] R. B. Rusu, Z. C. Marton, N. Blodow, M. Beetz, I. A. Systems, and T. U. München, "Persistent point feature histograms for 3d point clouds," in *Proceedings of the 10th International Conference on Intelligent Autonomous Systems (IAS-10)*, 2008.

[3] J. yves Bouguet, "Pyramidal implementation of the lucas kanade feature tracker," *Intel Corporation, Microprocessor Research Labs*, 2000.

[4] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[5] A. More, "Ice hockey equipment market growth, trends and forecasts (2020 - 2025)."

[6] M. Blinn, "The future of hockey: How the game looks now and how it is evolving."

[7] E. Hoerner, "The dynamic role played by the ice hockey stick. in safety in ice hockey,"

[8] T. Wu, D. Pearsall, and A. Hodges, "The performance of the ice hockey slap and wrist shots: the effects of stick construction and player skill.," *Sports Engineering*.

[9] A. Villaseñor, R. Turcotte, and D. Pearsall, "Recoil effect of the ice hockey stick during a slap shot.," *Journal of applied biomechanics vol. 22,3 (2006): 202-11.*

[10] W. E. Garrett and D. T. Kirkendall, *Exercise and Sport Science*. Lippincott Williams and Wilkins, 2000.

[11] K. Yücer, O. Wang, A. Sorkine-Hornung, and O. Sorkine-Hornung, "Reconstruction of articulated objects from a moving camera," in *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pp. 823–831, 2015.

[12] Z. Cai, H. Neher, K. Vats, D. A. Clausi, and J. Zelek, "Temporal hockey action recognition via pose and optical flows," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2543–2552, 2019.

[13] A. Goktepe, I. Ozfidan, H. Karabork, and F. Korkusuz, "Elbow but not knee joint kinematics can be assessed using photogrammetric methods during a non-stationary slap shot in ice hockey," *Scientific Research and essays*, 2010.

[14] Y. Michaud-Paquette, P. Magee, D. Pearsall, and R. Turcotte, "Whole-body predictors of wrist shot accuracy in ice hockey: a kinematic analysis," *Sports biomechanics*, 2011.

[15] R. Frayne, R. Dean, and T. Jenkyn, "Improving ice hockey slap shot analysis using three-dimensional optical motion capture: A pilot study determining the effects of a novel grip tape on slap shot performance," *Proceedings of the Institution of Mechanical Engineers, Part P: Journal of Sports Engineering and Technology*, vol. 229, pp. 136–144, 05 2014.

[16] B. Kays and L. Smith, "Effect of ice hockey stick stiffness on performance," *Sports Engineering*, vol. 20, pp. 1–10, 04 2017.

[17] M. Swarén, Q. Söhnlein, M. Holmberg, T. Stöggl, and G. Björklund, "Using 3d motion capture to analyse ice hockey shooting technique on ice," 06 2015.

[18] M. Salzmann and P. Fua, *Deformable Surface 3D Reconstruction from Monocular Images*, vol. 2. 09 2010.

[19] K. Yücer, O. Wang, A. Sorkine-Hornung, and O. Sorkine-Hornung, "Reconstruction of articulated objects from a moving camera," in *International Conference of Computer Vision Workshops (ICCVW)*, IEEE, 2015.

[20] C. Tomasi and T. Kanade, “Shape and motion from image streams under orthography: a factorization method,” *International Journal of Computer Vision*, vol. 9, pp. 137–154, 2004.

[21] C. Bregler, A. Hertzmann, and H. Biermann, “Recovering non-rigid 3d shape from image streams,” vol. 2, pp. 690 – 696 vol.2, 02 2000.

[22] W. Brand, “Morphable 3d models from video,” in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 2, pp. II–II, 2001.

[23] K. Nagano, J. Seo, J. Xing, L. Wei, Z. Li, S. Saito, A. Agarwal, J. Fursund, and H. Li, “Pagan: Real-time avatars using dynamic textures,” *ACM Trans. Graph.*, vol. 37, Dec. 2018.

[24] V. Blanz and T. Vetter, “A morphable model for the synthesis of 3d faces,” in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’99, (USA), p. 187–194, ACM Press/Addison-Wesley Publishing Co., 1999.

[25] K. Sakurada, W. Wang, N. Kawaguchi, and R. Nakamura, “Dense optical flow based change detection network robust to difference of camera viewpoints,” 2017.

[26] S. Parashar, D. Pizarro, A. Bartoli, and T. Collins, “As-rigid-as-possible volumetric shape-from-template,” pp. 891–899, 12 2015.

[27] M. Perriollat, R. Hartley, and A. Bartoli, “Monocular template-based reconstruction of inextensible surfaces,” *International Journal of Computer Vision*, vol. 95, pp. 124–137, 09 2008.

[28] T. Alldieck, M. Magnor, W. Xu, C. Theobalt, and G. Pons-Moll, “Video based reconstruction of 3d people models,” pp. 8387–8397, 06 2018.

[29] S. Zuffi, A. Kanazawa, and M. Black, “Lions and tigers and bears: Capturing non-rigid, 3d, articulated shape from image,” pp. 8387–8397, 06 2018.

[30] X. Han, H. Laga, and M. Bennamoun, “Image-based 3d object reconstruction: State-of-the-art and trends in the deep learning era,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1–1, 2019.

[31] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb, “State of the art on 3d reconstruction with rgb-d cameras,” *Computer Graphics Forum*, vol. 37, no. 2, pp. 625–652, 2018.

[32] Z. Li, A. Heyden, and M. Oskarsson, “Template based human pose and shape estimation from a single rgb-d image,” in *ICPRAM 2019 - Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods* (A. Fred, M. De Marsico, and G. S. di Baja, eds.), pp. 574–581, SciTePress, 2019.

[33] T. Yu, Z. Zheng, K. Guo, J. Zhao, Q. Dai, H. Li, G. Pons-Moll, and Y. Liu, “Doublefusion: Real-time capture of human performances with inner body shapes from a single depth sensor,” 2018.

[34] R. de Charette and S. Manitsaris, “3d reconstruction of deformable revolving object under heavy hand interaction,” 2019.

[35] R. B. Rusu, N. Blodow, and M. Beetz, “Fast point feature histograms (fpfh) for 3d registration,” in *Proceedings of the International Conference on Robotics and Automation (ICRA*, 2009.

[36] G. Farnebäck, “Two-frame motion estimation based on polynomial expansion,” in *Proceedings of the 13th Scandinavian conference on Image analysis*, 2003.

[37] M. A. Fischler and R. C. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” *Commun. ACM*, vol. 24, p. 381–395, June 1981.

[38] G. M. Y. Chen, “Object modeling by registration of multiple range images,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1991.

[39] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, and B. Chen, “L1-medial skeleton of point cloud,” *ACM Trans. Graph.*, vol. 32, July 2013.

[40] B. Bickel, M. Bächer, M. A. Otaduy, W. Matusik, H. Pfister, and M. Gross, “Capture and modeling of non-linear heterogeneous soft tissue,” in *ACM SIGGRAPH 2009 Papers*, SIGGRAPH ’09, (New York, NY, USA), Association for Computing Machinery, 2009.

[41] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.

[42] Q. Zhou, J. Park, and V. Koltun, “Open3d: A modern library for 3d data processing,” *CoRR*, vol. abs/1801.09847, 2018.