Tissue Tracking and Motion Compensation for Robotic Surgery

Kyle Lindgren

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Electrical Engineering

University of Washington

2017

Committee:

Blake Hannaford, Chair

Howard Chizeck

Program Authorized to Offer Degree:
Electrical Engineering

University of Washington

Abstract

Tissue Tracking and Motion Compensation
for Robotic Surgery

Kyle Lindgren

Chair of the Supervisory Committee:
Professor Blake Hannaford
Electrical Engineering

The ability to negate tissue motion without complex physical constraints would greatly benefit beating heart surgery, procedures where respiration causes tissue movement, and in the case of physically unstable operating environments such as a battlefield, moving vehicle, or space station. Using a calibrated stereo camera, it is possible to measure the 3D position of an object with relation to any known frame, and subsequently its motion. A method has been developed and implemented with the RAVEN^{TM1} surgical research robot for tracking and compensating for tissue motion with the goal of uncovering future research directions which will lead to real-time implementation of this technology. Results reveal insights regarding algorithm optimization, robot hardware limits, and the likelihood that the methods used are capable of operating in real-time with adequate computer hardware utilization.

---

[1]The RAVEN is a registered trademark of the University of Washington and Applied Dexterity.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

RAVEN: Surgical research robot designed at the University of Washington and University of California Santa Cruz. The RAVEN is currently manufactured and sold by Applied Dexterity.

END EFFECTOR: Tool connected to the end of a robot arm used to interact with the environment.

STEREO ENDOSCOPE: Surgical imaging instrument that utilizes two cameras with a known baseline, allowing for depth measurements while viewing the operating site.

FIELD OF VIEW: With respect to cameras, the FOV communicates the angle through which the camera sensor captures the environment.

FEATURE POINT: A distinguishable structure in an image.

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to his advisor, Blake Hannaford, and his fellow labmates for facilitating the many brainstorming sessions that proved invaluable to this project's contributions. Extensive gratitude is also due to Kevin Huang for always being available with an insightful perspective.

DEDICATION

to my family and friends who inspire me inside the lab, and out

Chapter 1

INTRODUCTION

1.1 Motivation

The area of surgical robotics affords many benefits over traditional surgical practices. Remote control of the robot allows a surgeon to operate from a great distance, potentially in a more secure location. Computationally powerful robots capable of executing surgeon actions with tools designed for minimally invasive surgery increase the number of surgeries capable of benefiting from a minimally invasive approach. Additionally, the hardware in place is capable of advancing the benefits of surgical robotics further, if provided with intelligent software that takes advantage of the available tools and current research practices. This project aims to further current literature and demonstrate the feasibility of a surgical robot to register beating heart motion and apply inverted control to robot end effectors, computationally canceling the observed motion (Figure 1.1[2]). As such, by taking current methods applied in this space beyond simulation with prerecorded videos, we also aim to uncover further considerations necessary when advancing tissue tracking and motion compensation schemes for clinical use.

Registering heart motion, the first step in the process, consists of two parts: tracking and modeling deformable tissue. The tracking portion requires selecting specific points on the tissue surface and identifying their location in each frame of a video sequence. Modeling the region of interest (ROI) deformable tissue is tasked with estimating the surface shape in each frame given the location of the tracked points. These tracked points are sparsely distributed in the ROI to represent the whole region being modeled. The modeling functions used often follow physical models which minimize a function representing bending energy and the alignment error between the current warped image and the initial reference image.

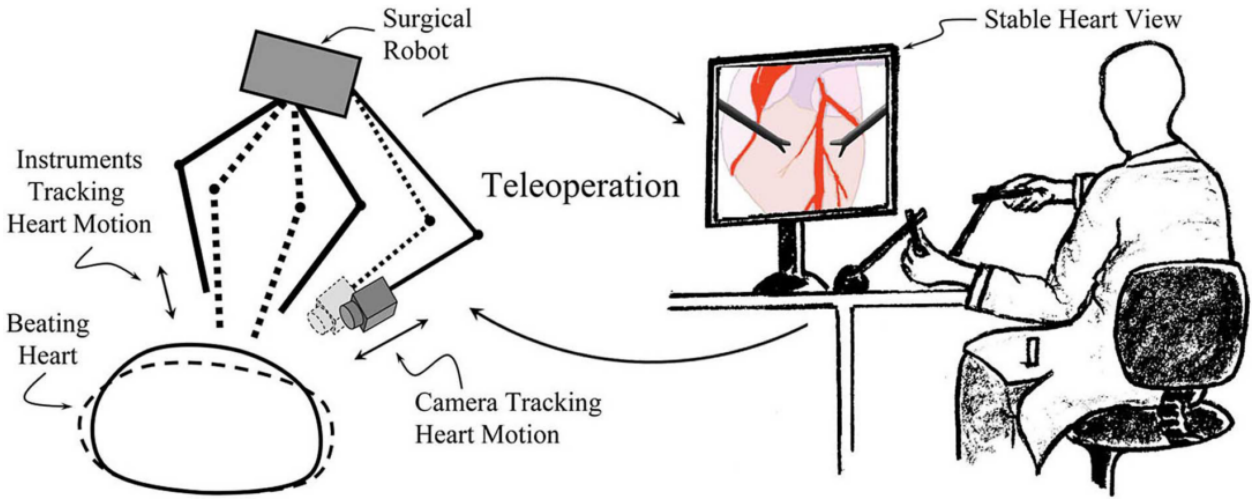

Figure 1.1: Overview of the surgical robotics tissue tracking and motion compensation scheme applied to beating heart surgery.

Three dimensional positioning of points in the ROI can then be found with accurate tracking of the same ROI in both cameras of a stereo endoscope. Setting the frame of reference for the tracked position to that of the robot provides simple end effector commands that compensate for ROI motion.

1.2 Literature Review

1.2.1 Surgical Robotics

Promoting laparoscopic surgery methods was one reason that drove early development of surgical robotics. The ability to allow surgeons to operate with small, dexterous tools through small incisions in the body cavity in a minimally invasive manner lowers patient recovery time. SRI International's development of a two-handed surgery unit for DARPA in the early 1990's [5] led to the creation of two companies with the goal of taking the technology to the civilian market. By 2001, Computer Motion Inc. (Goleta, CA) and Intuitive Surgical Inc. (ISI) (Silicon Valley, CA) both achieved FDA approved surgical robots, the Zeus and the da Vinci, respectively [3]. These two companies merged in 2003, keeping the ISI name, and

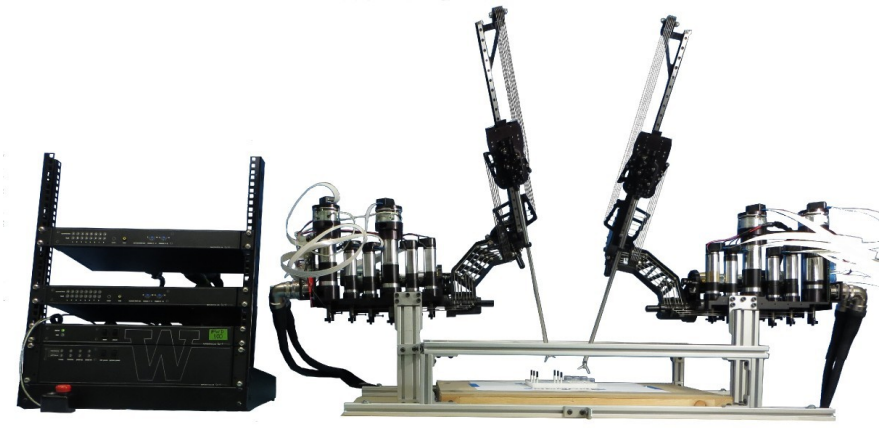

Figure 1.2: The RAVEN II surgical research robot. The control electronics are shown on the left with the cable-driven robot on the right.

have greatly advanced the field of surgical robots, with approximately 3660 da Vinci systems currently operating in hospitals across 64 countries.

To provide advancement to the field of surgical robots from academia, the University of Washington and the University of California Santa Cruz developed the RAVEN surgical research robot as an open source system. After confirming the feasibility of the design with the RAVEN I [7], development of the RAVEN II common research platform began. Seven research institutions located across the US formed the inaugural RAVEN community and received the first RAVEN II systems (Figure 1.2) in 2012 [4]. The RAVEN II platform has since expanded to approximately 18 globally located universities.

The RAVEN II's cable-driven arms consist of 3 spherical joints and a 4 Degree of Freedom tool. The RAVEN II operates on the popular Robot Operating System (ROS) to promote development from a wide audience and also support simple integration with peripheral systems also communicating via ROS. Real-time execution of the C++ source code occurs in a 1000 Hz control loop to maintain high fidelity control.

1.2.2 Tissue Tracking

There have been several implementations with different approaches applied in the area of tracking tissue. Matching of salient points was the focus of early work in [10], [15], and [16] with natural landmarks on the surface serving as the distinct tracking points. Such points are found in subsequent frames by applying similarity measures such as the Sum of Square Difference (SSD) and Normalized Cross Correlation (NCC) to tested image segments and the tracked point. These techniques require simple implementations but perform poorly in the case of rotation and scale change.

Feature tracking methods have steadily increased with more robust online learning and classification schemes developed in [19] and [9]. Menglong Ye et al. in [19] used a detection cascade scheme incorporating Haar-like features. The main benefit of Haar features is their use of integral images which increase calculation speed and maintain the same computation time for features of all sizes. [9] exploits a flaw with typical feature tracking methods in that the information encoded and the way it is represented is context specific. The dynamic appearance of tissue during surgery does not work well for ad hoc assumptions and therefore [9] proposed an online feature tracker which utilizes common feature tracking techniques while learning optimal representations for features.

1.2.3 Modeling Deformable Tissue

Unlike feature based matching, surface modeling can be incorporated with region based similarity measures and real-time implementations, a necessary requirement for the goal of aiding robotic beating heart surgery. In geometric surface modeling, a physical model approximates the tracked surface by minimizing the bending energy and the alignment error between the current warped image and the initial reference image.

Many physical models exist that can characterize soft tissue with free-form deformations (FFD) showing promising results in [17] and [6]. In [17], Stoyanov et al. used a piecewise bilinear map of the target region, accounting for vertical and horizontal motion. In this case,

minor inter-frame motion is found by minimizing the image difference between the current image and the reference image by warping with uniform image shifts. In [6], Lau et al. tracked a soft tissue region in a prerecorded beating pig heart video sequence using a B-spline physical model. A notable reason for the B-spline FFD model's excellent performance for the purpose of tracking soft tissue is its ability to account for the locations of every pixel in a region using a relatively minor set of parameters. Utilizing the positioning of only a few particular points, denoted control points, facilitates efficient computation and surface reconstruction.

Radial basis functions (RBFs), another surface model parameterization, also perform well in this application space as demonstrated by Rogério Richa with the thin-plate spline (TPS). In [13], Richa introduced the TPS for modeling heart surface deformations and tracking motion. Similarly to B-splines, the TPS models the reference image with a sparse set of control points which are tracked in each frame of a video sequence. Warping of current images is determined by the new positions of the control points in the current image and the TPS-defined dependence that pixels in the ROI region have with the control points. Region-based similarity measures serve as the feature tracking scheme to locate the control points in this implementation and will be discussed extensively in chapter 2. Richa showed favorable results when testing offline with a prerecorded video sequence of a beating pig's heart.

1.3 Project Goals and Hypothesis

In the reviewed literature, no tissue tracking and motion compensation systems were fully implemented on a surgical robot. This project proposes to integrate the necessary hardware and software tools to enable an end effector of the RAVEN II surgical robot to follow the 3D trajectory of a tissue point of interest as it moves according to a recorded calf heartbeat trajectory. Moreover, this project will study sped up computer vision template-based feature tracking, deformable surface modeling, and a method for creating an inexpensive stereo camera that mimics the performance of stereo endoscopes. The chapters that follow describe

the analysis of the problem, approach to the solution, and a discussion of the results.

1.4 Contributions

The contributions of this project are as follows:

- An operational tissue tracking and motion compensating system on a surgical robot.
- An inexpensive stereo camera with a 12.8 mm baseline that performs similarly to a stereo endoscope using only 3D printed parts, mirrors, and image processing commands.
- Considerations for future research directions targeted at implementing surgical robotic tissue tracking and motion compensation schemes in real-time.

Chapter 2

ANALYSIS

2.1 Stereoscopic Photography

A necessary component to tracking objects in 3D is the ability to measure depth, a task single cameras cannot natively do well. Adding a depth component to computer vision applications follows the same technique used by humans to perceive distance: adding a second camera. To see how this works, we first consider a pinhole camera, as shown in Figure 2.1. The pinhole camera represents a simplified model of a camera with the image plane at focus and hence the focal length f away from the center of projection O. We further generalize the pinhole example by moving the image plane to the opposite side of the center of projection, what would normally be the pinhole. A 3D point $P = (X, Y, Z)$ captured on the camera's image plane is shown at coordinate $P_c = (u, v)$. Using similar triangles and the focal length f, the map from the 3D point P to the 2D point P_c is defined as

$$\frac{f}{Z} = \frac{u}{X} = \frac{v}{Y}$$

which also gives

$$u = \frac{fX}{Z} \tag{2.1}$$

$$v = \frac{fY}{Z} \tag{2.2}$$

Just as human eyes are separated in the horizontal direction, most stereo cameras are fixed a certain distance apart in the horizontal direction. Although separating in the vertical direction exhibit similar performance, for simplicity we will demonstrate the depth problem for a horizontally separated stereo pair. Two cameras in this configuration capturing point P will register the same Y and Z measurements but different X values due to their fixed

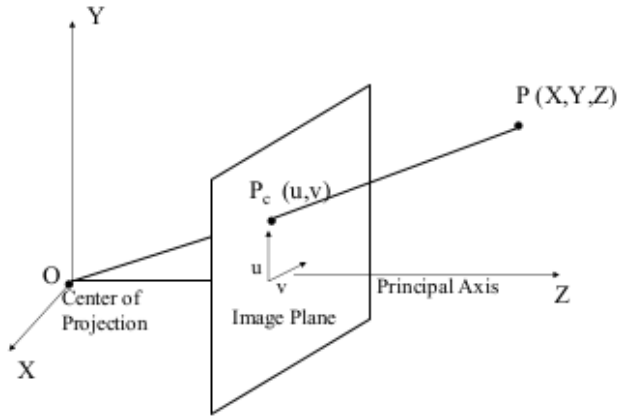

Figure 2.1: A further simplified pinhole camera with the image plane shown to the right of the center of projection. Normally in a pinhole camera the center of projection is shown as the pinhole with the image appearing inverted on the image plane located to the left of the pinhole. This version communicates the same concepts without inverting the image.

horizontal distance apart, denoted the *baseline*, B. As shown in equation 2.1, different X values will cause the u coordinates of P_c to also differ. For this case, we have

$$u_l = \frac{fX_l}{Z} \quad u_r = \frac{fX_r}{Z}$$

for the left and right cameras, respectively. The pixel-wise difference of u_l and u_r is known as the *disparity* between the same point captured by separate cameras, and communicates the depth as follows:

$$u_l - u_r = \frac{fX_l}{Z} - \frac{fX_r}{Z} = \frac{f(X_l - X_r)}{Z}$$

With the baseline defined as the horizontal distance between the two cameras, the baseline will also represent the horizontal difference between P due to both cameras measuring P according to their own base frame of reference. Accordingly, we get

$$B = X_l - X_r$$

and we can solve for the depth, Z:

$$u_l - u_r = \frac{fB}{Z}$$

Figure 2.2: The da Vinci surgical robot's approximately 8 mm and 5 mm baseline stereo endoscopes.

$$Z = \frac{fB}{u_l - u_r} \tag{2.3}$$

Note that disparity is inversely proportional to depth.

2.2 Reduced-Baseline Stereo Camera

As shown in equation 2.3, the distance from the camera to any point requires the point to appear in both camera image planes. The field of view (FOV) of the cameras set by the focal length and the baseline of the stereo configuration therefore determines the depth range, as objects too close will not appear in either or both cameras while objects far away push the disparity value down to zero. Accordingly, stereo endoscopy utilizes stereo configurations with small baselines to provide useful depth imagery inside a small surgical cavity. The da Vinci, for instance, uses endoscopes with baselines of approximately 8 mm and 5 mm, as shown in Figure 2.2.

While the cameras in Figure 2.2 perform very well, they also have a significant cost caused by the use of expensive miniaturized components. As such, we explored alternatives to minimize project costs and developed our own similarly performing stereo endoscope at

a fraction of the cost. Our design consists of two relatively inexpensive cameras mounted onto a 3D printed bracket also holding two mirrors. Figure 2.3 shows the design and our implementation of the pseudo stereo endoscope with both cameras facing towards mirrors set at a 45° angle with respect to each camera.

This unique design allows us to effectively create a stereo camera with the image axes of each camera running parallel after being redirected by the mirrors. The redirection pushes the image centers closer than the bodies of the cameras would allow (38 mm), achieving a baseline of only 11.82 mm. A few additional considerations necessary for proper performance of this stereo endoscope include:

- The mirrors must be *first-surface* mirrors as the commonly used *second-surface* mirrors have a layer of glass which protects the reflective surface but also causes refraction.

- The stereo setup must be fully calibrated after assembly to find the correction matrices which fix imperfections in camera alignment. Correct alignment allows us to use equation 2.3 to calculate depth because we can assume points seen in both cameras will always have the same vertical pixel coordinates, $v_l = v_r$.

- Lastly, mirrors flip images, so they must be programmatically flipped again before image processing begins.

2.2.1 Design Validation

The reduced-baseline stereo camera needed to be thoroughly validated as the entire system relies on accurate position estimates from control point pixel locations and disparities. Accordingly, intended device operation and common lab materials led to the development of the following tools and testing methods.

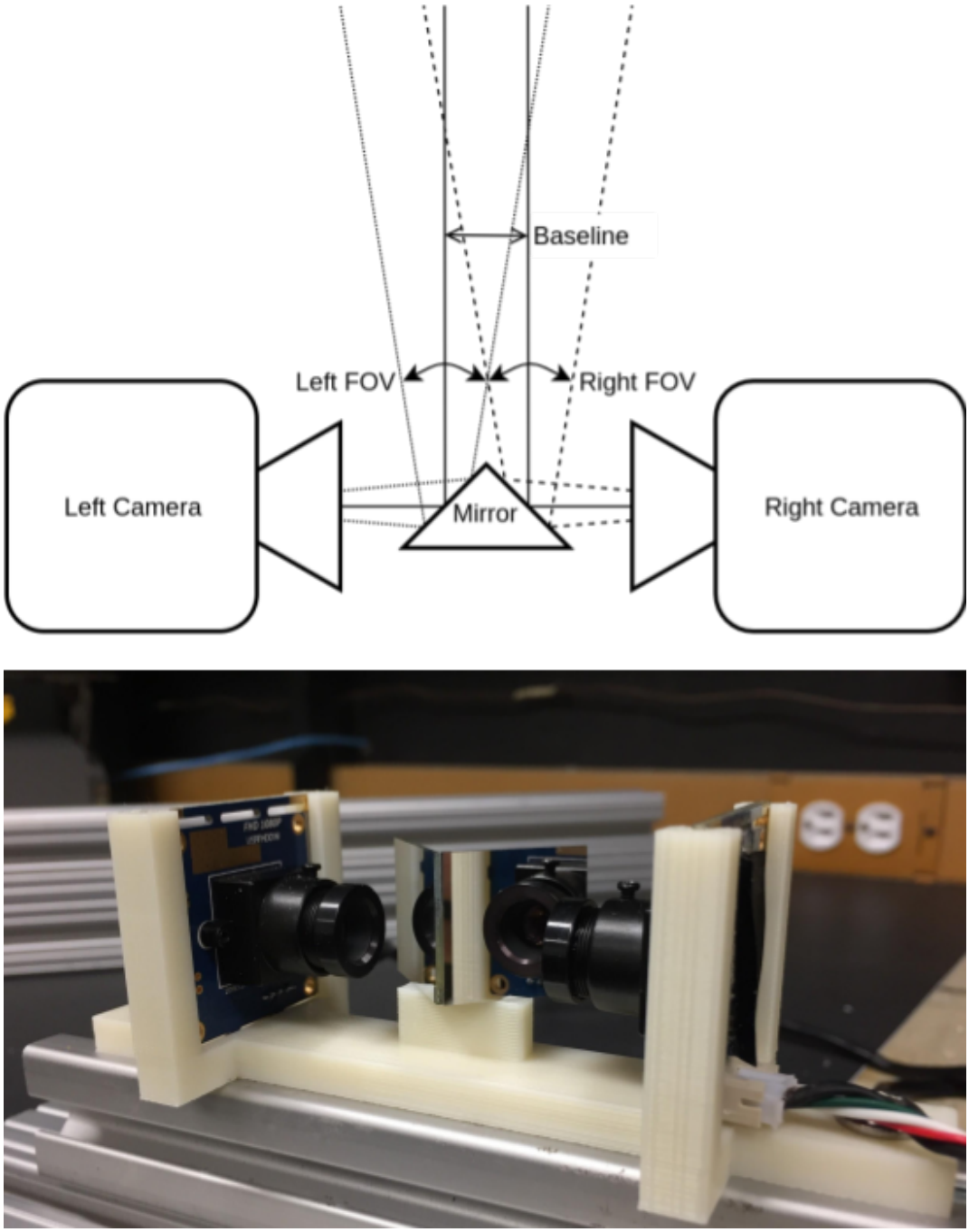

Figure 2.3: Top: Top view of the stereo camera design with mirrors redirecting camera viewpoints, reducing the baseline. Bottom: Implementation of the stereo endoscope with an 11.82 mm baseline.

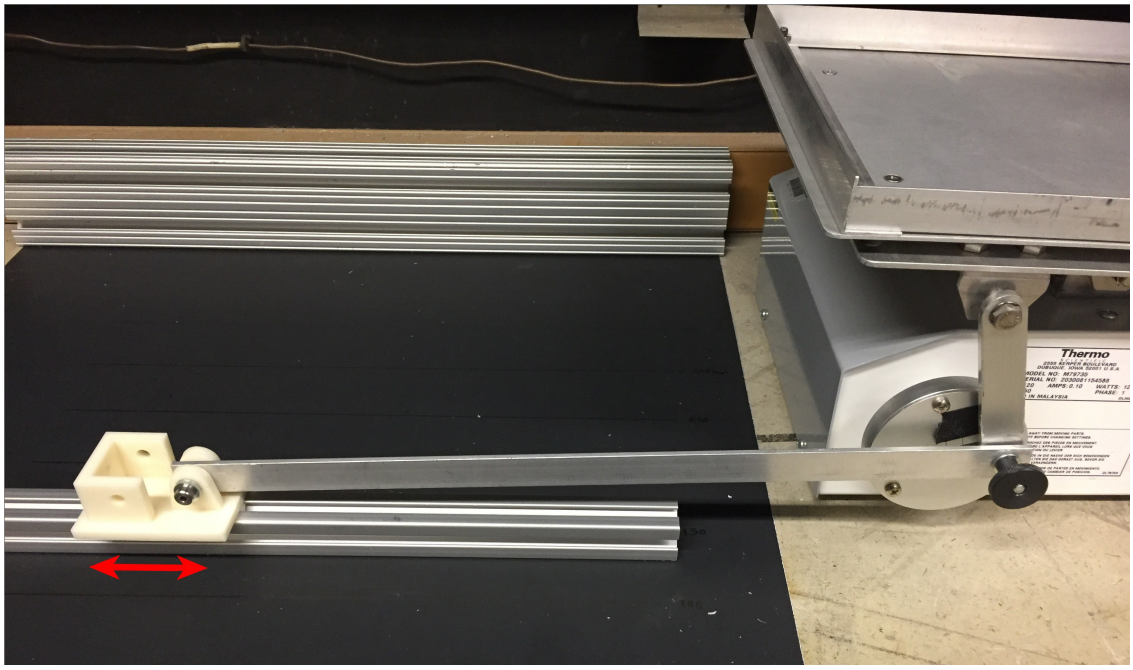

Figure 2.4: Originally designed to mix solutions, the rocker table's rotation mechanism was utilized to provide sinusoidal motion to the tracked object affixed to an 80/20 guide.

Tools

Stereo camera position measurement performance on sinusoidal trajectories was studied as the results may provide meaningful assessment given the sinusoidal nature of common motion trajectories caused by physiological factors. A Barnstead Thermolyne Platform Vari Mix rocker table (Figure 2.4, right) was used to provide the sinusoidal motions with a frequency determined by a device setting and amplitude by connection with the table's rotation mechanism. Rocker tables are intended for use in wet labs for mixing solutions but the exposed rotation mechanism allowed for repurposing. An 80/20 guide provided a fixed linear path with motion generated from a beam pushing and pulling driven by the table's rotating actuator.

Additionally, the 80/20 guide and 3D printed cart (Figure 2.4) were used in linear tests, providing a fixed path and device for holding an object to be tracked.

Methods

The performance of the stereo camera tracking objects in 3D was assessed separately for each dimension. Each dimension went through measurement testing with both a stepping method, where the tracked object moved in fixed step size increments, as well as a sinusoidal method.

In the stepping method, the stereo camera remained stationary while an object was manually moved by small increments along one dimension. An 80/20 guide provided the fixed path for the object with measurements along the side marking increments. The guide was placed perpendicularly to the camera and at several distances within the camera's operating space for the X and Y tests. The Z tests were conducted with the tracked object moving along the image centerline. Recorded videos were then processed with position measurements written to a file for analysis. Matlab was used to load the resulting files and plot the results to be compared with ground truth measurements.

The sinusoidal motion method made use of the rocker table test setup by moving with fixed amplitudes and frequencies along the guide. Similarly to the stepping method, tests were conducted at several depths for the X and Y dimensions and the Z dimension was also tested through multiple depth ranges. Data collection and analysis was performed with the same methodology as with the stepping method.

Results

Figure 2.5 shows the results from the stepping method testing for both the 10 mm and 20 mm step sizes. Figure 2.6 shows a sample of the results for the sinusoidal method testing for each dimension, characteristic of the results for the other depths and amplitudes tested. Sinusoid measurements were captured at four separate depths for the X and Y dimensions and three for the Z dimension with only a subset of results shown in the interest of space and readability. Z dimension results are placed at the midpoint of the tested range. For instance, results from the 30 mm sine amplitude testing between depths of 210 mm and 240

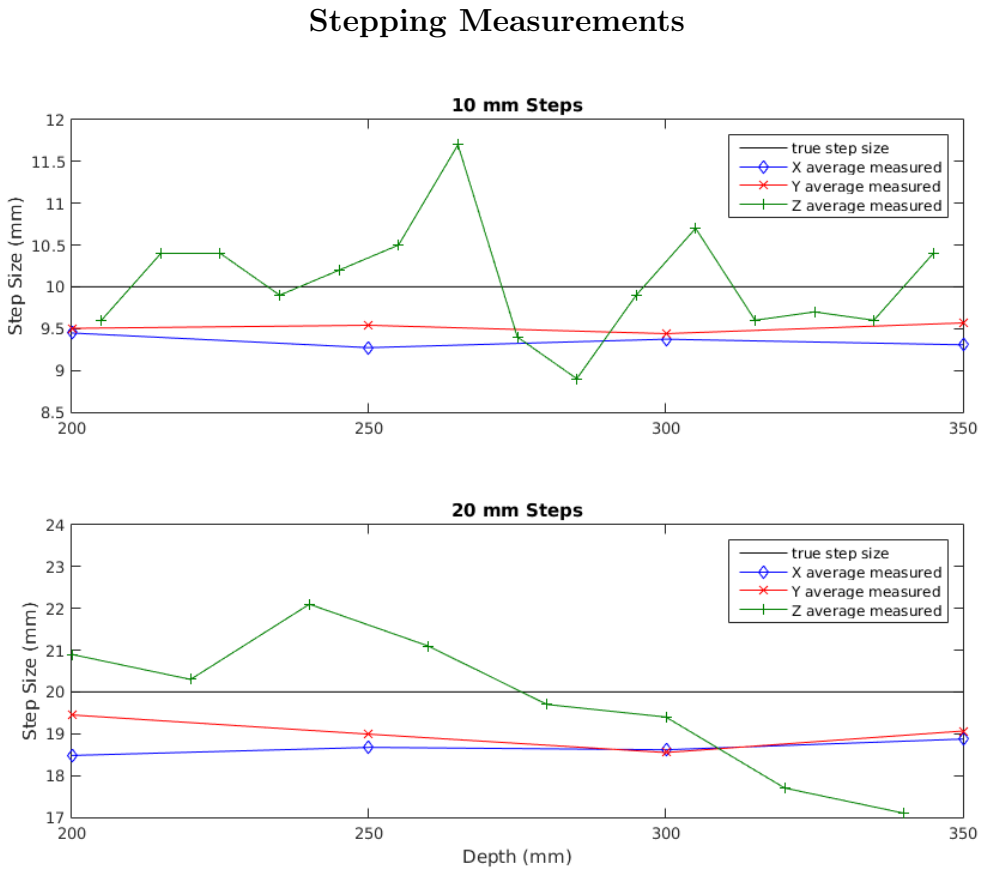

Figure 2.5: Stepping measurements with each point representing the average measured step size in all tests for the selected depth and dimension. Black lines mark ground truth values.

mm were placed at the 225 mm depth value in the following plots.

With sinusoidal motion provided by the rocker table, there was not a method for recording ground truth data for the exact sinusoid trajectory and therefore the black lines shown in Figure 2.6 are only intended for reference. Accordingly, only the amplitudes of the waves were used for assessing measurement performance. The black lines are sine waves with amplitudes equal to the magnitude tested and frequencies manually tuned to match the recorded data.

Additionally, Figures 2.7 and 2.8 summarize the observed errors from both testing methods, with percentage as well as magnitude errors shown.

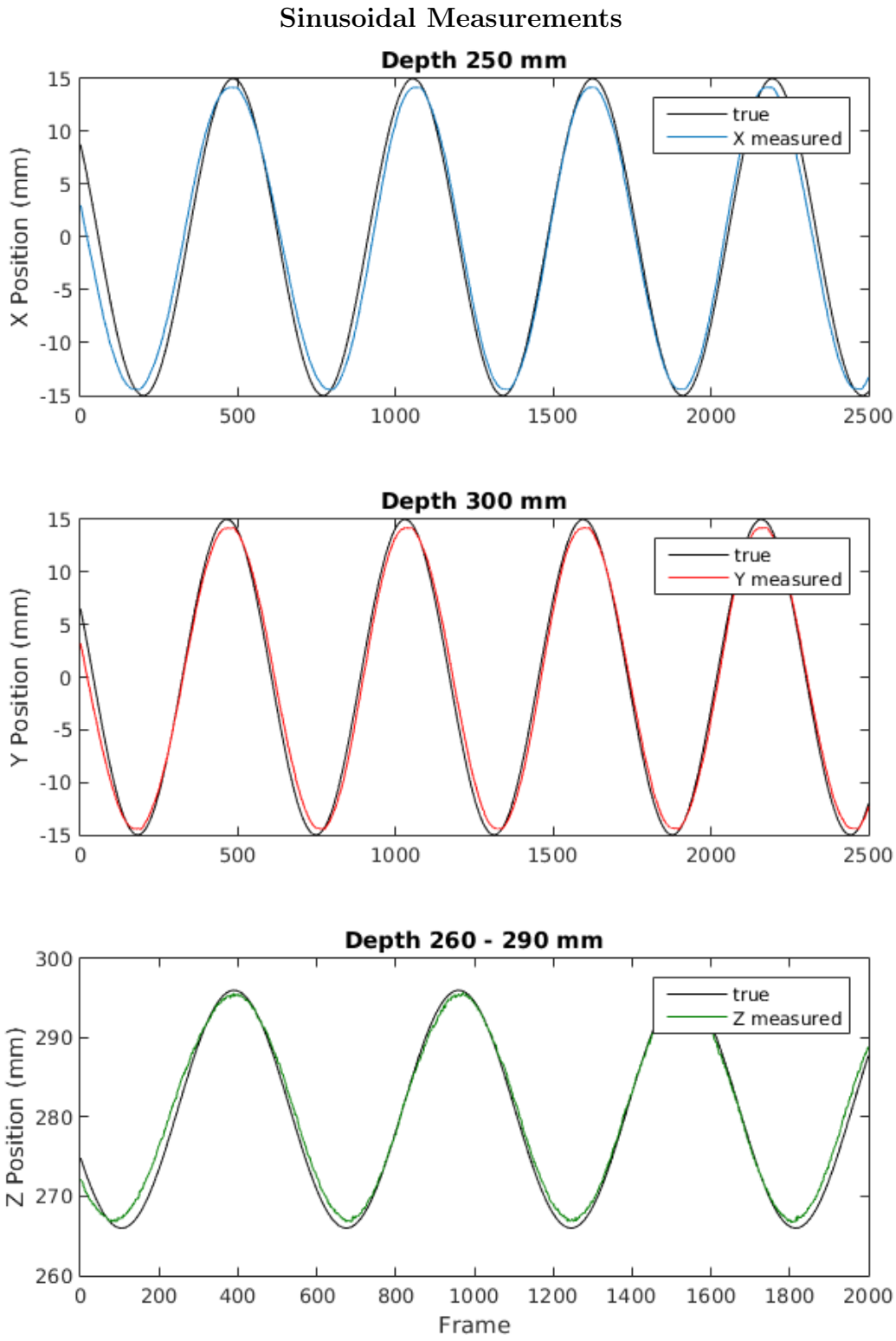

Figure 2.6: Sample plots showing results from the sinusoidal testing for each of the dimensions. Black lines loosely represent the path of the tracked object during testing but are not ground truth values.

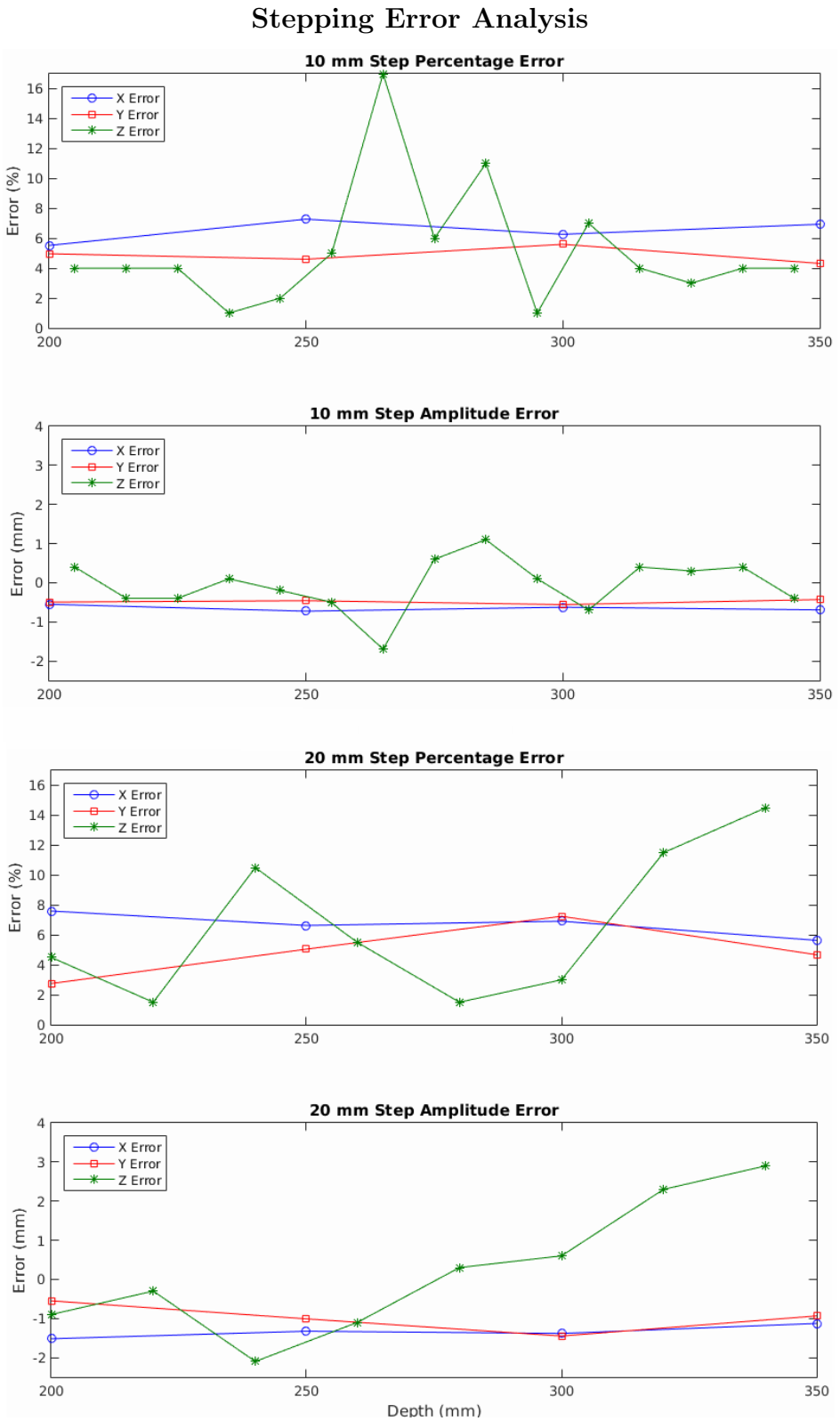

Figure 2.7: Average error shown as percent and absolute values for the 10 mm (top two) and 20 mm (bottom two) step testing.

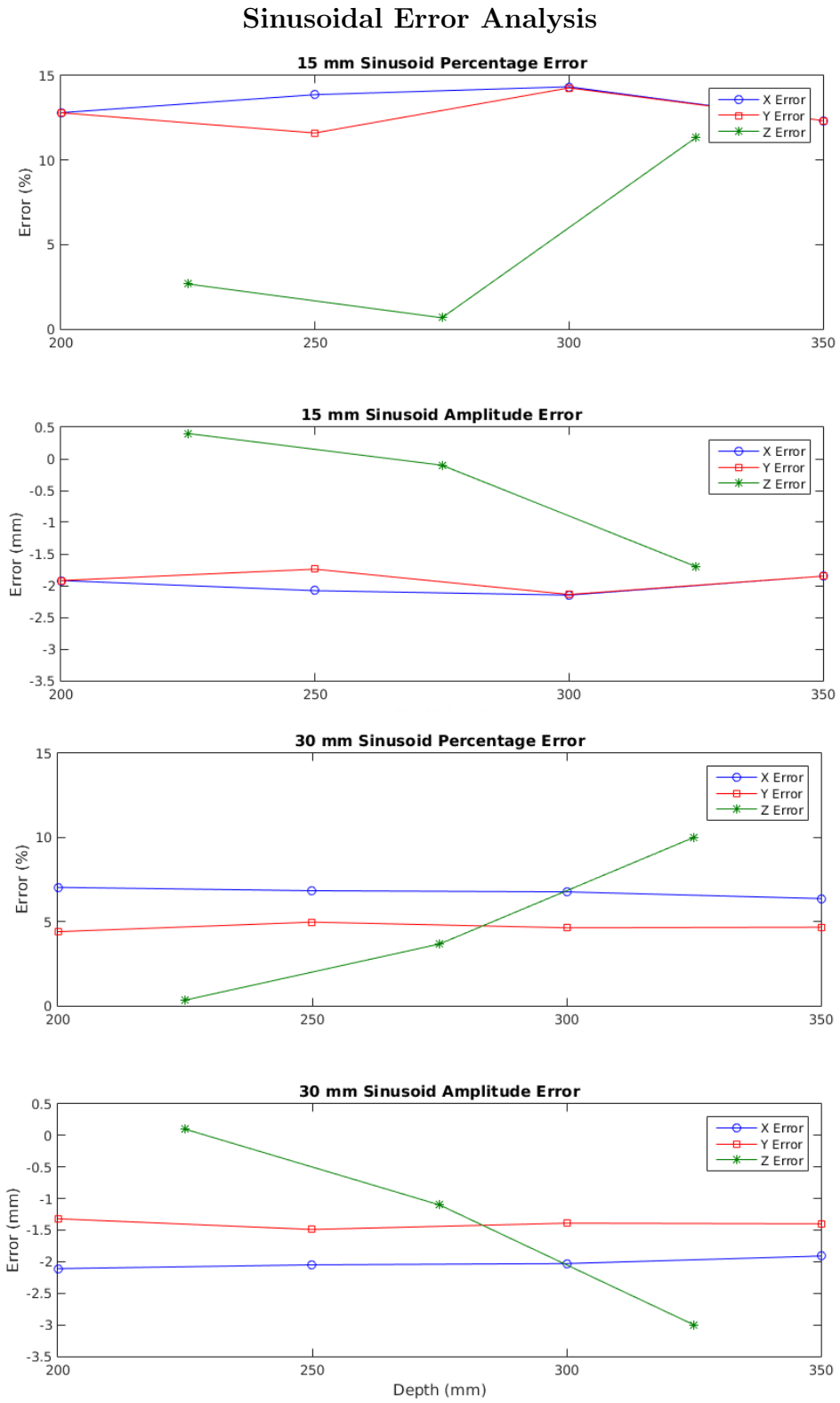

Figure 2.8: Average error shown as percent and absolute values for the 15 mm (top two) and 30 mm (bottom two) sinusoidal testing.

Conclusions

Although the results show relatively low accuracy (Figure 2.9), further analysis indicates high precision (Table 2.1) through the calculated standard deviations of errors. As such, we attribute the low accuracy to calibration error and bracket deformation. The reduced-baseline stereo camera was calibrated using OpenCV supported functions for the commonly used method of finding camera parameters with a checkerboard of known dimensions viewed in a variety of configurations. This is a proven method but far from perfect when conducted outside of a professional setting, especially when using a checkerboard of such small size as we were, appropriate for our depth range. Additionally, the 3D printed bracket inevitably experienced minor changes in shape after calibration, magnifying already present inaccuracies. Future iterations will likely adopt a design more robust to bending, with considerable attention given to the portions holding each camera.

Although the reduced-baseline stereo camera exhibited less than ideal accuracy through the testing methods, high precision indicates successful design and also considerations for future versions. Overall, the accuracy was determined to be sufficient for use in our application, and the performance to cost ratio ideal for prototyping before making large investments in research equipment.

Stepping Error STD (mm)

	Amplitude	
Dimension	Small	Large
X	0.078	0.163
Y	0.055	0.369
Z	0.669	1.703

Sinusoid Error STD (mm)

	Amplitude	
Dimension	Small	Large
X	0.139	0.084
Y	0.169	0.070
Z	1.097	1.563

Table 2.1: Standard deviations of measurement error for the stepping method (left) and sinusoidal method (right). Values are averaged across all tested depths for the given dimension and amplitude size.

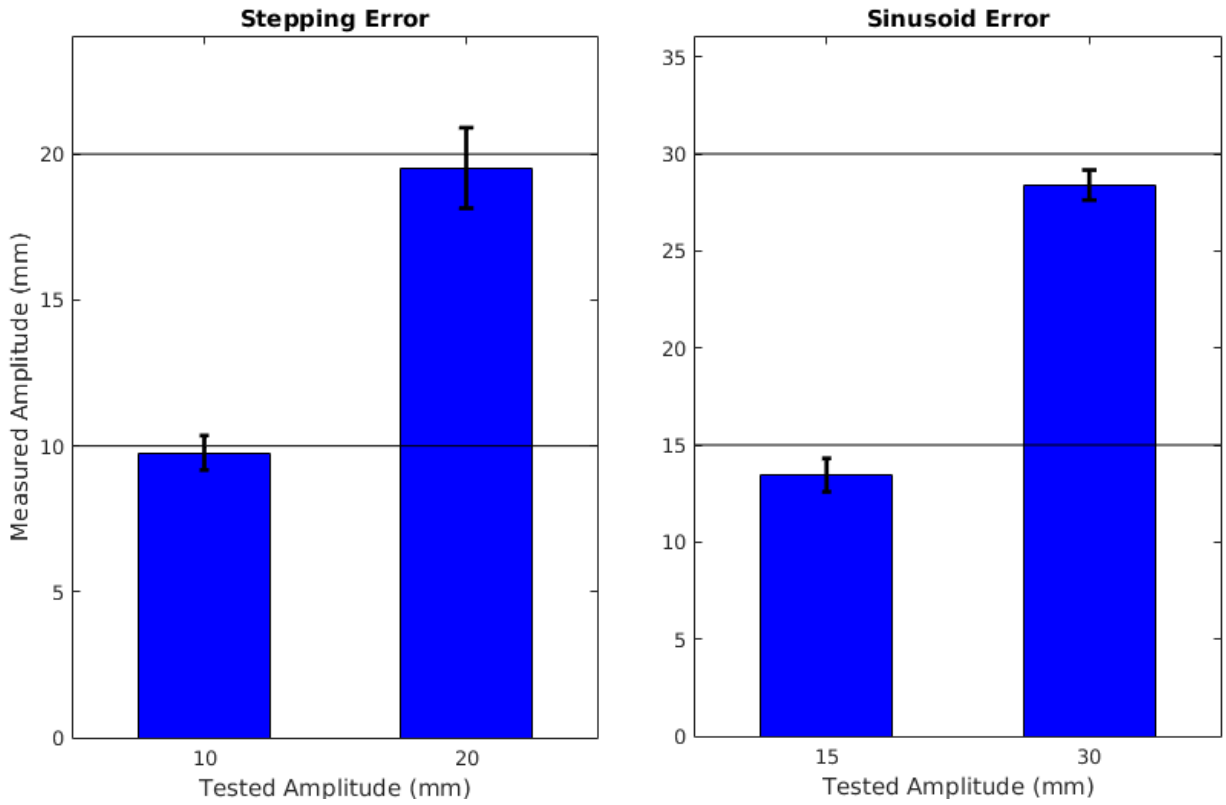

Figure 2.9: Summarized error analysis averaged across all 3 dimensions, shown for both methods and tested amplitudes with error bars representing standard deviations. Horizontal black lines indicate ground truth measurements.

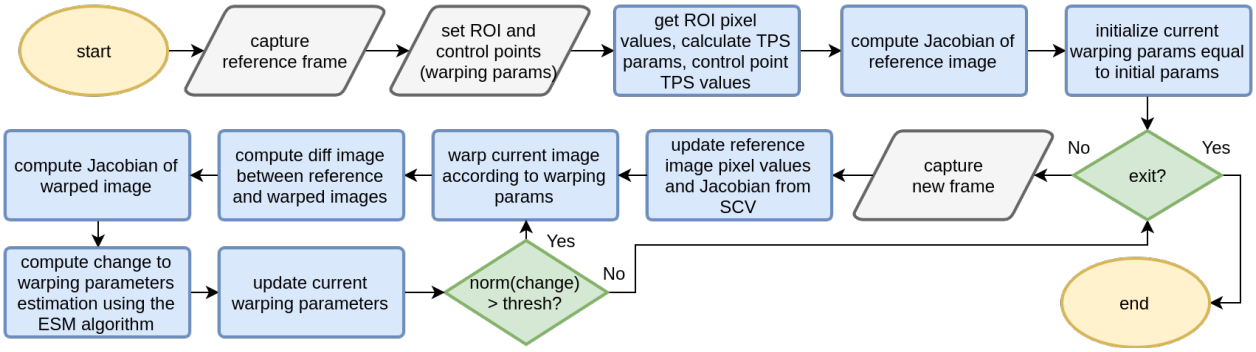

Figure 2.10: Flow chart of the tissue tracking and modeling process.

2.3 Tracking and Modeling Tissue

Tracking and modeling a region of interest in the surgical site consists of a few key components, which will be detailed in this section. The process begins with control points, unique points located within the region of interest at coordinates with strong features. Control points are estimated in each frame and used to back-warp the current image to align with the reference image. Accurate alignment confirms accurate estimation of the control point locations and therefore the shape of the warped region. Back warping estimation is performed with the thin-plate spine (TPS) radial basis function while control point locations in new frames are iteratively estimated utilizing template based searching and the efficient second-order minimization (ESM) algorithm. A flow chart detailing the tissue tracking and modeling steps is shown in Figure 2.10.

2.3.1 Thin-Plate Spline

Much of the TPS modeling is taken from Richa in [12] and [13]. This section presents the TPS derivation as it is computed and used in our application.

The TPS radial basis function allows us to model the entire region of interest while only tracking the position of a few key points. This is achieved by approximating pixel coordinates

that minimize the bending energy:

$$E_f = \int\int_{\Re^2} (f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2) dxdy$$

with f_{xx}, f_{xy}, and f_{yy} representing second partial derivatives of f. The function $f(\mathbf{x})$ for region of interest point $\mathbf{x} = (x, y)$ is defined as:

$$f(\mathbf{x}) = r_1 + r_2 x + r_3 y + \sum_{i=1}^{n} w_i U(\|\mathbf{c}_i - \mathbf{x}\|)$$

with TPS basis function $U(s) = s^2 log(s)$, coordinates of control points c_i, and warping parameter vectors r, w. Because the TPS is a $\mathbb{R}^2 \rightarrow \mathbb{R}$ mapping, we need two functions f^x and f^y to achieve x and y pixel coordinates of the region of interest. Given that these functions share control points, we can create a stacked function m such that:

$$m(\mathbf{x}) = \begin{bmatrix} f^x \\ f^y \end{bmatrix} = \begin{bmatrix} r_2^x & r_3^x & r_1^x \\ r_2^y & r_3^y & r_1^y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} + \sum_{i=1}^{n} \begin{bmatrix} w_i^x \\ w_i^y \end{bmatrix} U(\|\mathbf{c}_j - \mathbf{x}\|)$$

With $t = (w_1, \ldots, w_n, r_1, r_2, r_3)^T$, the parameter vectors $\mathbf{t}^x$ and $\mathbf{t}^y$ that define f^x and f^y based on control point coordinates can be solved with the following linear system:

$$\begin{bmatrix} \mathbf{L} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{O} \end{bmatrix} \begin{bmatrix} \mathbf{t}^x & \mathbf{t}^y \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{P}} \\ \mathbf{0} \end{bmatrix}$$

where $\mathbf{L}_{ij} = U(\|\mathbf{c}_j - \mathbf{c}_i\|)$, $\mathbf{P}$ contains the pixel coordinates of the control points in the reference frame as $(1, x, y)$, $\mathbf{O}$ and $\mathbf{0}$ are 3×3 and 3×2 zero matrices respectively, and $\hat{\mathbf{P}}$ is a matrix of control point estimated pixel coordinates. Denoting the leftmost matrix as $\mathbf{K}$, the parameter vectors $\mathbf{t}^x$ and $\mathbf{t}^y$ can be solved for as:

$$\begin{pmatrix} \mathbf{t}^x & \mathbf{t}^y \end{pmatrix} = \mathbf{K}^{-1} \begin{bmatrix} \hat{\mathbf{P}} \\ \mathbf{0} \end{bmatrix}$$

Utilizing the equation for the parameter vectors, we can solve for region of interest pixel coordinates according to control point estimations. The matrix of estimated coordinates

$\mathbf{P'} = \left[\mathbf{x'_1}, \mathbf{x'_2}, \ldots, \mathbf{x'_q}^T\right]$ of q region of interest pixels can be computed by:

$$\mathbf{P'} = \begin{bmatrix}\mathbf{V} & \mathbf{W}\end{bmatrix} \mathbf{K}_* \hat{\mathbf{P}}$$

where $\mathbf{V}_{ij} = U(\|\mathbf{x}_j - \mathbf{c}_i\|)$, $\mathbf{W}_i = (1, x_j, y_j)$, and $\mathbf{K}_*$ is the $(n+3) \times n$ sub-matrix of $\mathbf{K}^{-1}$. Denoting $[\mathbf{V}\ \mathbf{W}]$ as $\mathbf{M}$ and the vector of control point coordinates as $\mathbf{h} = (\mathbf{h}^{xT}, \mathbf{h}^{yT})^T$ formed from $\hat{\mathbf{P}} = (\mathbf{h}^x, \mathbf{h}^y)$, we can define the warping function for pixel $\mathbf{x}_i$ in ROI $\mathbf{x}$ as:

$$w(\mathbf{x}_i, \mathbf{h}) = \begin{bmatrix}\mathbf{x'_i} & \mathbf{y'_i}\end{bmatrix} = [\mathbf{M}_i \mathbf{K}_* \mathbf{h}^x \mid \mathbf{M}_i \mathbf{K}_* \mathbf{h}^y] \tag{2.4}$$

2.3.2 Efficient Second-Order Minimization

Control points provide the connection between the reference frame and subsequent frames. Knowing where all of the control points are in a frame allows us to find the position of any region of interest pixel in that frame using the TPS and warping according to equation 2.4. Feature tracking is a common method to locate control points because it performs with high accuracy, however, searching for and matching features is computationally expensive and not suitable with the goal to operate in real-time. Template based searching using the efficient second-order minimization (ESM) algorithm is much faster by measuring control point coordinate estimate accuracy using whole region pixel value errors (difference in ROI pixel values in the warped and reference images) and updating control point estimates according to the image gradient.

According to the ESM implementation in [13], the region of interest coordinates from the current image are warped according to the TPS algorithm and control point locations and compared with the reference image to get the pixel-wise error image. The error image along with the image gradient of both the reference image and current image are used to update control point estimates. Control point estimates are iteratively updated in this fashion towards the location that minimizes the error image magnitude. Convergence is accepted when the update to the control point estimates is below a threshold. This method is adopted from [8] and this section presents a summary of the ESM derivation.

The ESM algorithm achieves second-order convergence with a computational cost of the same order as a first-order method by using Taylor series expansions to eliminate computation of the Hessian matrix. When solving for the control point coordinates $\mathbf{h}$ that minimize the error between the warped image and the reference image, we are trying to minimize

$$\mathbf{F}(\mathbf{h}) = \sum_{\mathbf{x}\in\mathbf{A}} [\mathbf{I}(w(\mathbf{x},\mathbf{h})) - \mathbf{T}(\mathbf{x})]^2 = \|\mathbf{f}(\mathbf{h})\|^2$$

with the current image $\mathbf{I}$, reference image $\mathbf{T}$, pixel coordinate $\mathbf{x}$ in the ROI, and $w(\mathbf{x},\ \mathbf{h})$ our TPS warping function with control points $\mathbf{h}$ mapping pixel $\mathbf{x}$ from the reference image $\mathbf{T}(\mathbf{x})$ to the current image $\mathbf{I}$. To find a local or global minimum of the function above, we must find a point h_* such that $[\nabla_h F]_{\mathbf{h}=h_*} = 0$, where ∇_h is the gradient operator with respect to $\mathbf{h}$. A second-order Taylor series of $\mathbf{f}$ about $\mathbf{h} = \mathbf{0}$ gives

$$\mathbf{f}(\mathbf{h}) = \mathbf{f}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\ \mathbf{h} + \frac{1}{2}\mathbf{M}(\mathbf{0},\mathbf{h})\mathbf{h} + \mathcal{R}(\|\mathbf{h}\|^3)$$

where $\mathbf{J}(0) = [\nabla_h f]_{h=0},\ \mathbf{M}(z,h) = [\nabla_h \mathbf{J}]_{h=z}h$, and $\mathcal{R}(\|\mathbf{h}\|^\ni)$ is the third-order remainder. Similarly, the Taylor series of the Jacobian $\mathbf{J}$ about $\mathbf{h} = \mathbf{0}$ is

$$\mathbf{J}(\mathbf{h}) = \mathbf{J}(\mathbf{0}) + \mathbf{M}(\mathbf{0},\mathbf{h}) + \mathcal{R}(\|\mathbf{h}\|^3)$$

Combining the two equations above leads to

$$\mathbf{f}(\mathbf{h}) = \mathbf{f}(\mathbf{0}) + \frac{1}{2}(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\mathbf{h}))\ \mathbf{h} + \mathcal{R}(\|\mathbf{h}\|^3)$$

As $\mathbf{h}_* \approx \mathbf{0}$, a second-order approximation of $\mathbf{f}$ in $\mathbf{h}_*$ is

$$\mathbf{f}(\mathbf{h}_*) \approx \mathbf{f}(\mathbf{0}) + \frac{1}{2}(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\mathbf{h}_*))\ \mathbf{h}_*$$

Through further approximations at the solution, $\mathbf{f}(\mathbf{h}_*) = \mathbf{0}$, the ESM algorithm substitutes $\mathbf{J}(\mathbf{0}) + \mathbf{J}(\mathbf{h}_*)$, the Jacobians of the difference image, with $(\mathbf{J_I} + \mathbf{J_T})$, the Jacobians of the current image and reference image respectively. Substituting $(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\mathbf{h}_*))$ with $(\mathbf{J_I} + \mathbf{J_T})$, at the solution, we get

$$\mathbf{h}_* = -\left(\frac{\mathbf{J_I} + \mathbf{J_T}}{2}\right)^+ \mathbf{f}(\mathbf{0}) \tag{2.5}$$

In our application, $\mathbf{h}_*$ is a vector representing the increment to the warping parameters $\mathbf{h}$ which minimizes the error image between the reference and warped current image. Equation 2.5 is iterated updating the warping parameters until the magnitude of the update falls below a threshold.

2.3.3 Lighting Correction

Visual tracking in computer vision applications requires consideration of varying environment lighting. Lighting correction is critically important for this application due to the incredibly variable lighting exhibited in surgical cavities as well as the ESM algorithm's performance reliant on comparison of images taken at different times. Following [14], we have implemented the sum of conditional variance (SCV) similarity measure to account for changes in lighting. The SCV fits this application well due to its invariance to non-linear illumination variations and minimal computation expense.

Similarly to the ESM notation, we consider the current image $\mathbf{I}$ and the reference image $\mathbf{T}$ both $(n \times m)$ matrices. Let pixel coordinate $\mathbf{x} = (\mathbf{x}, \mathbf{y}) \in \{\mathbf{1}, \mathbf{2}, \ \ldots, \mathbf{m}\} \times \{\mathbf{1}, \mathbf{2}, \ \ldots, \mathbf{n}\}$ and $w(\mathbf{x}, \ \mathbf{h})$ be our warping function with parameters $\mathbf{h}$ mapping pixel $\mathbf{x}$ from the reference image $\mathbf{T}(\mathbf{x})$ to the current image $\mathbf{I}$. We also let $\mathbf{I} \in [0, d_{\mathbf{I}}]$ and $\mathbf{T} \in [0, d_{\mathbf{T}}]$, with $d_{\mathbf{I}}$ and $d_{\mathbf{T}}$ representing the maximum discrete pixel intensity values that images $\mathbf{I}$ and $\mathbf{T}$ can hold. Using

$$\hat{\mathbf{T}}(\mathbf{x}) = \mathscr{E}(\mathbf{I}(w(\mathbf{x}, \ \mathbf{h})) \mid \mathbf{T}(\mathbf{x}))$$

with $\mathscr{E}(\cdot)$ being the expectation operator, we get

$$SCV(\mathbf{h}) = \sum_{\mathbf{x}} (\mathbf{I}(w(\mathbf{x}, \ \mathbf{h})) - \hat{\mathbf{T}}(\mathbf{x}))^2 \tag{2.6}$$

The image $\hat{\mathbf{T}}$ is found from the joint intensity distribution P between the two images. P is a $(d_{\mathbf{T}} \times d_{\mathbf{I}})$ matrix with elements (i, j) representing the probability of the intensity co-occurrence $(\mathbf{I}(w(\mathbf{x}, \ \mathbf{h})) = i, \ \mathbf{T}(\mathbf{x}) = j)$ for a given pixel and warping parameters. The joint

intensity distribution is calculated as

$$P_{ij} = \frac{1}{p} \sum_{\mathbf{x}} \phi(\mathbf{I}(w(\mathbf{x},\ \mathbf{h})) - i)\phi(\mathbf{T}(\mathbf{x}) - j)$$

where $p = n \cdot m$, $\phi(s) = 1$ for $s = 0$, $\phi(s) = 0$ otherwise and $i \in [0, d_{\mathbf{I}}]$ and $j \in [0, d_{\mathbf{T}}]$. And the conditional expectation can be computed as:

$$\mathscr{E}(\mathbf{I}(w(\mathbf{x},\ \mathbf{h})) \mid \mathbf{T}(\mathbf{x})) = \frac{\sum_i i \cdot P_{ij}(i,\ \mathbf{T}(\mathbf{x}))}{\sum_i P_{ij}(i,\ \mathbf{T}(\mathbf{x}))}$$

At the beginning of each iteration, we calculate $\hat{\mathbf{T}}$ for all pixel values to represent the reference image and account for illumination variations since capturing the initial $\mathbf{T}$ reference image.

2.3.4 Extracting 3D

The previous derivations consider only a single camera system. In our application with the reduced-baseline stereo camera, we process video streams separately but in parallel which creates two sets of parameters used for virtually all of the calculations including the control points. To support 3D measurements, however, we must ensure the control points both cameras are tracking represent the same real-world points, which we force during the initialization sequence.

During initialization, the region of interest selected in the left camera's reference image is used with the whole right camera reference image to find matching feature points. The SURF feature detector [1], chosen for its robustness and fast calculation, returns a predefined number of control points that match in both images. A new region of interest must be selected in the case that insufficient features are found.

Accurate estimates of the control points in each camera allows us to calculate the 3D position of any pixel in the region of interest according to

$$P_p = C \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = C^{-1} P_p \tag{2.7}$$

where $[XYZ]^T$ is the 3D coordinates of the point, C is the camera calibration matrix, and P_p is the point in projective coordinates. We arbitrarily choose the left camera to represent the world frame so C is the left (3×3) intrinsic camera calibration matrix and the point $[XYZ]^T$ is with respect to the location of the left camera.

Point $x = (u_l, v_l)$ as seen in the left camera image plane is converted to projective coordinates following $P_p = (su, sv, s)$ where s is the depth component calculated from equation 2.3. By left multiplying with the inverse of the calibration matrix, we can find the 3D position of any point in the region of interest.

2.4 Summary

An overview of stereoscopic photography was presented with a derivation for acquiring depth values. Stereo endoscopes require very small baselines due to the small environments they operate in. While stereo endoscopes used during real surgeries are expensive, a novel design and implementation was shown for a reduced-baseline camera that mimics the performance of real stereo endoscopes at a fraction of the cost.

Algorithms for tracking and modeling tissue have also been presented with the TPS serving as the physical model and the ESM as the fast control point estimator used with template-based feature tracking. Accounting for lighting changes must also be considered during operation and the SCV similarity measure was presented as an efficient yet robust compensator.

Chapter 3

METHODS

3.1 Proposed Architecture

The system consists of three primary software modules: scene capture, remote computation, and RAVEN II surgical robot action. The stereo cameras capture the surgical site, sending a video stream to the remote computer through a wired USB connection. The remote computer processes images to locate the control points and calculate the 3D position of the point of interest (POI), the specific point in the ROI the robot is following, sending the position as a ROS message [11] containing an incremental move command to the robot via network connection. Using inverse kinematics, the robot updates end effector positions accordingly.

3.1.1 Scene Capture

As detailed in Chapter 2, a novel stereo camera with a reduced-baseline served as a pseudo stereo endoscope to capture the surgical site. OV2710 camera modules were chosen for their wide range of supported resolutions and support for USB connection. In the interest of minimizing computational load, we used a relatively low resolution of 640x480. We replaced the default 3.6 mm lenses with 12 mm lenses as a longer focal length is more appropriate for our operating range and narrow field of view (FOV). The unique design of the stereo camera's mirror redirection imposes a constraint on the range of baselines supported. Placing a camera too close to the mirror causes the camera's lens to obstruct the redirected FOV, and too far away forces part of the FOV off the mirror, as can be reasoned in Figure 2.3. The narrower the FOV, the closer the cameras can be placed to the tip of the mirror platform, where the effective baseline is smallest.

Each camera sends JPEG compressed image data through USB 2.0 connections to the

Camera Specs

Sensor	OV2710
Sensor Size	1/2.7 inches
Resolution	640x480
Pixel Size	3 μm x 3 μm
Frame Rate	varies

Table 3.1: Specifications for the cameras used in the reduced-baseline stereo camera. Note the frame rate is dependent on program execution time with frames being pulled from the camera on demand.

remote computer. Commands to pull a new image from the stream occur after control point estimations have converged in the previous frame.

3.1.2 Remote Computation

The remote computer implemented the majority of the functionality in this project by managing the stereo camera interface, performing image processing, and facilitating communication with the robot. The computer ran Ubuntu 16.04 on an Intel 2.10GHz Core i7-3612QM CPU with 8GB of 1333MHz DDR3 SDRAM. The C++ code was developed in the Qt IDE heavily utilizing the computer vision OpenCV 3.2 libraries for matrix storage and common image processing operations. OpenCV was chosen for its well established and supported functionality, which eased development.

Upon starting the program, the remote computer turns on the stereo cameras and begins the program initialization process of loading configuration parameters, finding and syncing control points, and setting algorithm constants. After initialization completes, the program then enters a while loop, tracking the ROI in new images until stopped by the user. Updated POI locations are sent via ROS messages to the RAVEN II at the end of each while loop iteration.

Initialization

At the beginning of initialization, the calibrated camera matrices for each camera must be loaded to rectify offsets and distortions after capturing new images. These matrices provide correction to radial and tangential distortion as well as misalignments that would normally break the stereo camera assumptions that allow us to calculate depth using only horizontal pixel disparities. The inverse transpose of the left calibration matrix is also calculated here for later use converting projective coordinates to 3D.

Next, each camera captures what is referred to in the rest of the program as the *reference* frame, **T**. The operator then selects a point in the left reference frame indicating the desired ROI. A predefined region size is fit around this point and control points are placed on features found in both left and right reference images, as determined by the SURF detector. The SURF detector searches the left ROI and the entire right reference frame for the predefined number of matching points. If insufficient features are found or all of the returned points group in only a subset of the ROI, the operator must choose a new region in the left reference frame. For this work, an 80x80 pixel sized region was chosen as the fixed size for the ROI as a compromise between providing enough area to find good control points and not creating unnecessarily large matrices that required time consuming calculations. Similarly, the number of control points was chosen to be four to provide sufficient tracking performance while minimizing computation.

Knowing the locations of the control points then allows computation of the fixed matrices used in the TPS algorithm. These matrices only require computation once as the control points do not change during execution.

Tracking

The tracking portion of the remote computer processing occurs in a while loop beginning with new images requested from the stereo camera. These images receive the same calibration correction treatment as the reference images before being used to update the reference

images with the SCV algorithm. Next, a *do while* loop performs TPS and ESM calculations estimating the new location of the control points until the magnitude of the update falls below 10^{-2} pixels or a maximum of 30 iterations have occurred. In either case, convergence is accepted. The newly found locations of the control points are then used to calculate the 3D position of the POI. The POI position delta from the previous frame is sent to the RAVEN computer before returning to the top of the while loop and requesting new images.

ROS Communication

ROS serves as the communication platform sending POI position updates to the RAVEN computer via network connection. Both the remote computer and RAVEN computer run ROS Kinetic. Defining the RAVEN computer's URI as `ROS_MASTER_URI` on the remote computer allows the C++ project contained in a ROS catkin package to publish messages to a ROS topic created and subscribed to by the RAVEN.

The ROS topic `raven_automove` subscribed to by the RAVEN control software is used to send commands contained in messages of type `raven_automove` that hold the following ROS types and values:

Header	hdr
int32[6]	del_pos
geometry_msgs/Transform[2]	tf_incr

Our applications uses the `geometry_msgs/Transform[2] tf_incr` to transport relevant position data. The two element array supports separate commands for each arm but we sent identical data, opting to have both arms compensated in the same fashion. Though the RAVEN operates at 1000 Hz, we only send new `raven_automove` messages when a new position increment has been found.

Matrix Pseudoinverse Test

Matrix Library	ROI Pixel Size		
	2304	6400	12544
OpenCV	3.66 ms	7.94 ms	14.95 ms
Armadillo	2.85 ms	5.98 ms	10.72 ms
Eigen	22.82 ms	49.21 ms	92.41 ms

Table 3.2: Matrix pseudoinverse test using three matrix libraries with varying ROI sizes. Times shown are calculated as the average frame computation time across a 1000 frame video, without including conversion time.

3.1.3 Robot Action

After powering on the RAVEN, the computer creates several ROS topics to support ROS communication for sending and receiving commands, as well as sharing robot state information. Upon completing its homing process, in which the RAVEN initializes all of its joints with known values, the RAVEN subscribes to the `raven_automove` ROS topic. At 1000 Hz, the robot checks this topic for new messages, immediately acting according to received commands. The `raven_automove` topic utilizes inverse kinematics to find joint angles which solve requested end effector position increments contained in the received `raven_automove` message.

3.2 Statistics

The ESM algorithm requires computing the pseudoinverse of a dense matrix two times the number of control points wide by the number of ROI pixels tall in each iteration while determining control point locations. This costly operation led to exploration of linear algebra libraries that might outperform OpenCV. As shown in Table 3.2, the Armadillo linear algebra library computes pseudoinverses faster than OpenCV and Eigen for all tested ROI sizes. Also

Figure 3.1: Test setup used during the testing procedure with a Kuka youBot arm holding a tissue surface within the workspace of both the RAVEN and reduced-baseline stereo camera.

of note, conversion between OpenCV matrices and Armadillo matrices uses data storage locations instead of copying by value. Because of this, conversion only adds approximately 10% to the Armadillo pseudoinverse computation time.

3.3 Test Setup

A test setup (Figure 3.1) was developed to assess the tissue tracking performance while also revealing insights into motion compensation considerations through general observations.

3.3.1 Kuka youBot

The arm of a Kuka youBot (Figure 3.2) was used to simulate motion in the workspace of the RAVEN with the system tracking the youBot's end effector location. Utilizing an attached

Figure 3.2: Arm of the Kuka youBot.

wireless router, the youBot's onboard computer communicated with a separate ROS master computer, both running Ubuntu 12.04 and ROS Groovy. ROS masters provide naming and registration services to the rest of devices operating in the ROS environment. Using software written by a fellow labmate, Kevin Huang, the master computer ran the initialization process for the robot, starting a ROS node that utilizes inverse kinematics to actuate the end effector to absolute positions. The master computer was also used to connect via `ssh` to the youBot's computer to run our ROS node that published the absolute positioning of desired testing trajectories.

3.3.2 Trajectory

The Kuka youBot utilizes beating calf heart trajectory data (Figure 3.3) acquired by Erdem Tuna et al. [18]. The trajectory data was recorded from the left anterior descending (LAD) coronary artery at a rate of 249.5 Hz using a sonomicrometer crystal sutured on the cardiac

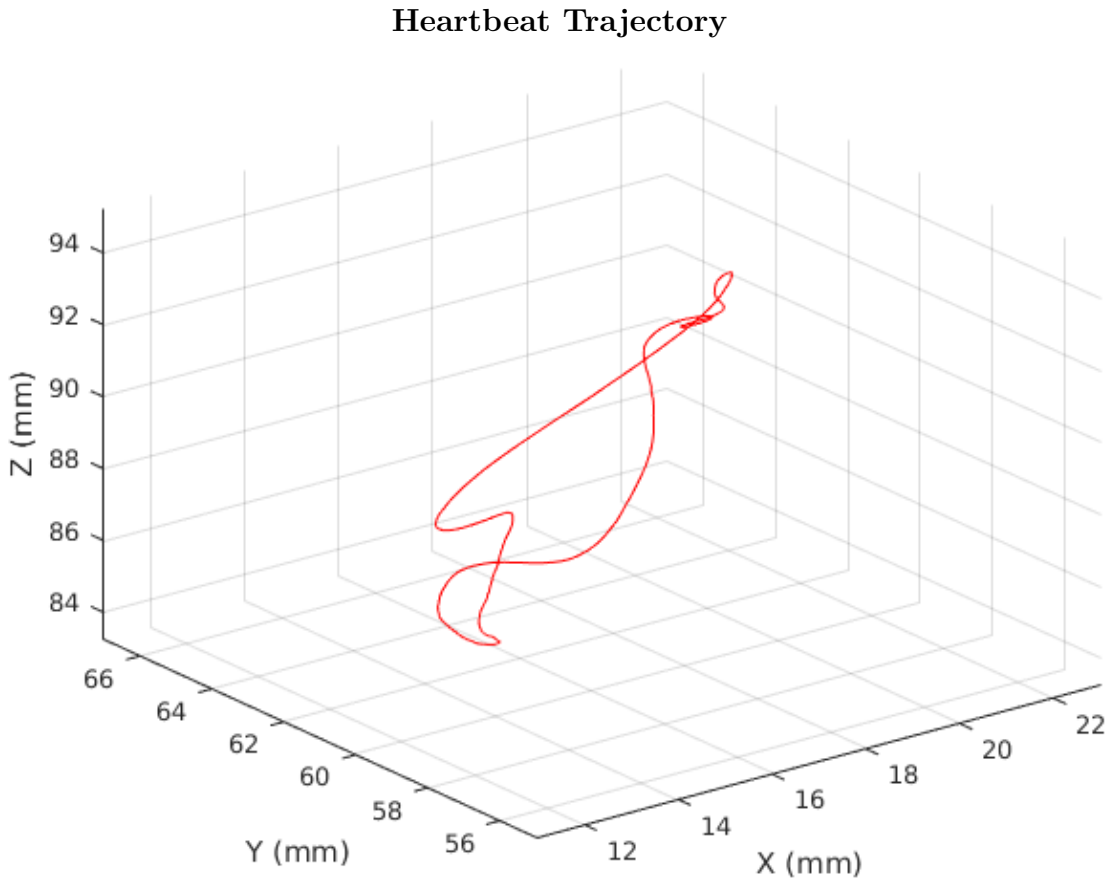

Figure 3.3: Perspective view of the calf heartbeat trajectory used in the test setup.

surface. The single heartbeat has range of (10.4 mm, 7.8 mm, 8.0 mm) in the respective dimensions.

3.3.3 Tracked Object

The Kuka youBot test setup provided the platform by which an artificial tissue object (Figure 3.4) copied the rigid body motion of the beating heart. The artificial tissue object is an image of organic tissue containing blood veins of many sizes and was chosen to represent organic tissue with high levels of detailed texture.

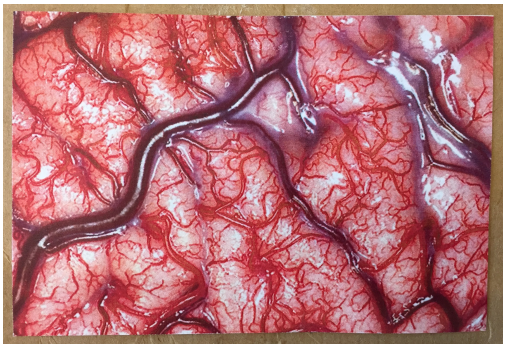

Figure 3.4: The artificial tissue image chosen as the object for tracking.

3.4 Testing Procedure

3.4.1 Objective

A testing procedure was developed to assess performance with the fully integrated system as well as relative computation expenses for various tissue tracking algorithm parameters. More specifically, the test setup specified above was used to find limits at which tissue tracking failed due to inter-frame computation time taking too long, allowing control point movement to become too great. Conducting these tests also revealed limiting factors of the system as well as those with more robustness.

3.4.2 Independent Variables

The size of the ROI and number of tracked control points varied during testing. The ROI was always a square shape with sizes differing in side length by 8 pixels. The number of control points started at three, increasing for the tested ROI size until tracking failure.

These two variables were chosen as independent variables because each directly correlates to a meaningful aspect of the system. The size of the ROI determines the surface the surgical operator can work in while receiving motion compensation. The number of control points determines the reliability of motion compensation in areas not directly located on a control point since control point positions are known during tracking with the other pixels in the

ROI estimated from control point estimates. More control points therefore increases position estimation of non-control point pixels in the ROI.

3.4.3 Dependent Variables

The tested trajectory, tracked surface, and ESM convergence constraints were kept constant throughout all testing. The heart trajectory (Figure 3.3) played in a continuous loop, repeating the same heartbeat. To allow for a wide range of tested ROI sizes and number of control points, the speed of the trajectory was kept at one-eighth its recorded speed, with a max velocity of 6.4 mm/s. The same tracked surface (Figure 3.4) was used in all testing as well with the same general region chosen as the ROI. Lastly, the threshold for control point estimation convergence (Figure 2.10) and max number of iterations were also kept constant. The threshold value was set to $1e^{-2}$ and the max number of iterations set to 30. Setting a limit on the iterations occasionally helps tracking when convergence stalls due to control point estimates bouncing back and forth between their true points. The limit accepts minor estimation error with the goal that convergence be reached in the next frame if minimal time is wasted.

3.4.4 Procedure

The test setup shown in Figure 3.1 was used to assess tissue tracking performance while varying the ROI size and number of control points. Testing began with a fixed ROI size and three control points, the minimal number necessary for image stabilization through affine transformations. The number of tracked control points was increased if the system could successfully track the ROI through ten heartbeats. Ten was observed to be sufficiently long enough so that if ten were achieved, the system could run indefinitely without accumulation of tracking errors that led to failure. Figure 3.5 shows examples of when tracking is running successfully and when tracking breaks down.

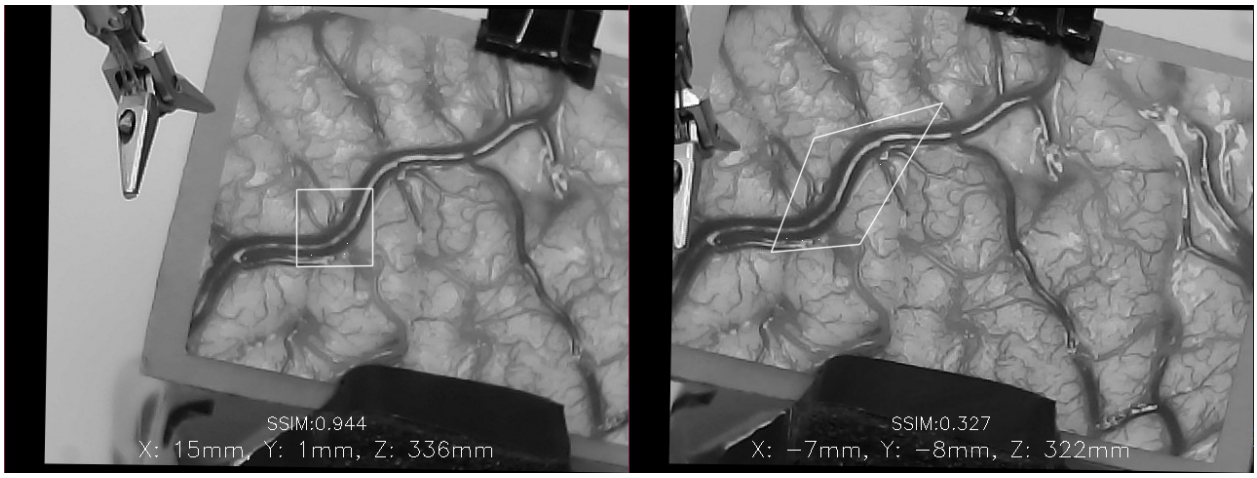

Figure 3.5: Successful tracking (left) is observed as correct placement of the white ROI bounding box as the tracked object moves. Failed tracking (right) is observed when control point estimates deviate far from their true locations, causing the ROI bounding box to warp.

Chapter 4

RESULTS AND EVALUATION

The minimum ROI size tested was 48×48 pixels as smaller sizes proved increasingly difficult to find sufficient numbers of control points with the tested tissue object. The maximum size that successfully tracked at least three control points through ten heartbeats was 168×168 pixels. With a video resolution of 640×480, the largest tested ROI occupied more than 9% of the frame.

It is also worth noting that tracking performance was observed to be improved when control points uniformly covered the ROI as opposed to bunching up in a small area. Control points are placed according to the SURF feature detector finding and matching points with high texture in both the left and right camera views during the initialization process without regard for placement of neighboring control points. Occasionally during testing, the ROI needed to be reselected to get control points more evenly spaced.

Results are evaluated according to the maximum ROI size that supports a given number of control points. For instance, if several ROI sizes support a maximum of n control points, it can be assumed that the computation time is relatively similar for each ROI size. Therefore, the largest ROI size should be chosen to maximize the motion compensating region without substantial computation increase. The overall maximum number of control points that can be tracked is also worth noting, indicating the optimal ROI size for the most accurate 3D measurements.

Results are shown in Figure 4.1 from finding the maximum number of control points that can be tracked starting with the minimum ROI size that supports sufficient control points up to the largest ROI that successfully tracks at least three control points.

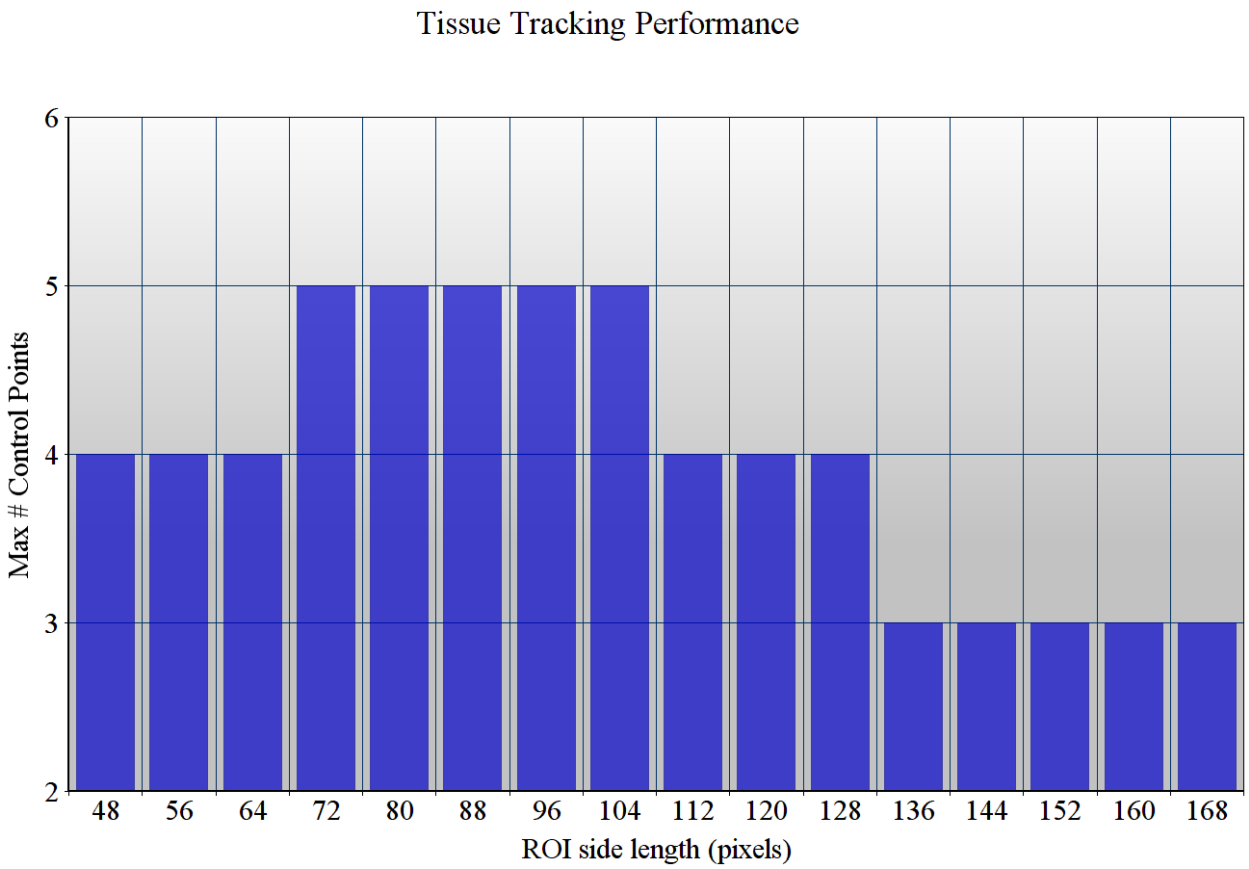

Figure 4.1: The results from tissue tracking testing, finding the maximum number of control points the system could track for a given square ROI size.

Chapter 5

DISCUSSION

Testing the system with the setup and procedure described in chapter 3 revealed many conclusions outside of those directly resulting from the testing methods. These insights relate to tracking shortcomings and robot hardware limits in addition to optimal tracking ROI size and number of control points.

The early stages of determining the heartbeat playback speed in the testing procedure revealed the control point tracking methods used have a high sensitivity to high accelerations. Vibration of the tracked object quickly caused tracking failure at playback speeds greater than one-eighth of the recorded calf heartbeat rate. The issue is caused by control points moving too far between frames and would subsequently benefit greatly from decreased time intervals between processed frames. This issue was exacerbated by the Kuka youBot's end effector which was observed to not maintain a perfectly steady position throughout the heartbeat trajectory. Controlling the youBot consisted of sending absolute position commands for the last joint, with the last link programmed to always remain parallel with the ground. The youBot's ability to keep the tracked object steady with respect to the last joint was not perfectly robust, but any injected errors to the trajectory were present throughout all testing.

Perhaps the most interesting results that came directly from the testing procedure (Figure 4.1) are that in some cases, a larger ROI can support more control points. These results contradict the assumption that increasing the ROI and/or number of control points incrementally increases computation time and therefore decreases tracking performance. Instead, we see adding pixels can aid control point estimation convergence and there is a balance where the ROI size either adds or lessens computation. Further study of the iterative up-

date to control point estimation using the ESM algorithm (Equation 2.5) shows that larger tracked regions provide more information relating to control point locations. This is evident by seeing that the Jacobians of the reference image and current image in addition to the difference image for each pixel in the ROI contributes to the control point update. Collectively, the added information has the potential to drive control point estimates to a converged solution with less iterations.

Operating the tissue tracking and motion compensation system also indirectly revealed RAVEN hardware limits which are also likely problematic for other surgical robot platforms. The main issue encountered involved reaching software defined current limits on some of the joints while sending motion commands to the RAVEN. In these cases, safety constraints kept the end effector from moving in unison with the tracked object despite successful tracking. While the current limits are thoughtfully in place to protect expensive and difficult to replace motors, they are often set according to steady state limits. Following this restriction blocks instances of high current that present little danger to the motors if maintained for only short bursts. Additionally, current limits seemed to be observed somewhat inconsistently during testing with the RAVEN software e-stopping in only certain regions of the workspace while following the same trajectory. This phenomena can be explained by the varying amount of force required to move the robot's end effectors depending on where they are located in the operating workspace.

As mentioned in the testing procedure, our implementation of the tissue tracking and motion compensation scheme operated at one-eighth real-time when following the heart trajectory. While this is quite far from real-time, it seems the methods used are feasible for real-time implementations given proper hardware and optimization techniques. The computationally intensive task of tissue tracking was executed on a relatively outdated laptop running on an Intel 2.10GHz Core i7-3612QM CPU with 8GB of 1333MHz DDR3 SDRAM. Additionally, several aspects of the software were optimized for the computer vision library used as well as the matrix library but GPU utilization was not explored. Therefore, using a robust system consisting of a current generation CPU and/or GPU, it is likely real-time

implementation of the stated methods are presently achievable, or will be in the near future.

5.1 Summary

From these results, the following assertions can be made about the tested tissue tracking and motion compensation scheme for robotic surgery:

- The thin-plate spline surface modeling function combined with the efficient second-order minimization algorithm for object tracking using control points is sensitive to motion between frames, even when the motion only exists across a few frames.

- In some cases, a larger ROI can support more control points. This is likely caused by the insight ROI pixels provide to control point estimation updates using the ESM algorithm since all pixels in the ROI contribute to the estimate.

- The RAVEN, and likely other surgical robot platforms, includes software defined current limits which may shut down the system when trying to move end effectors too quickly. Experiences will vary depending on the operating location in the robot's workspace due to torque requirements varying with joint configurations.

- The presented methods with our chosen computing hardware is far from real-time, but achieving real-time seems plausible with current generation processing power and/or GPU utilization, or at least will be with processor improvements in the near future.

Chapter 6

CONCLUSIONS

A tissue tracking and motion compensation scheme has been analyzed, implemented, and tested on the RAVEN surgical research robot. A novel stereo camera design with a reduced baseline was described and shown to perform adequately while costing much less than high performance stereo endoscopes used during surgeries today. The thin-plate spline algorithm provided surface modeling of a tracked tissue surface image. To enable surface modeling, feature points, denoted control points, on the surface needed to be found in each frame and searching was conducted using a region based method utilizing the efficient second-order minimization technique to achieve second-order convergence with the computational cost of first-order methods. Accurate tracking of the user specified ROI allows motion compensation control to be sent to the surgical robot for any point within the tracked region. This is accomplished with both cameras of the reduced-baseline stereo camera tracking the ROI and therefore knowing the real-world 3D position of all points in the region. Observed movements are then sent to the robot as incremental position commands to negate motion, moving the end effectors with the same observed motion of the ROI.

After validating project components separately, tests were conducted to evaluate performance of the tissue tracking methods in a live scenario tracking an object through a heartbeat trajectory, while also analyzing how the system performed fully operating on a surgical robot platform. Tests revealed the tissue tracking methods used are noticeably sensitive to motion, even when the motion exists only momentarily. It was also discovered that increasing the tracked region can potentially improve tracking performance by supporting additional tracking points. This surprising effect is likely due to faster convergence through the control point update step that utilizes ROI image Jacobians and the error image to drive

control point estimates towards their true position. With respect to performance of the robot, we found that current limits set to protect motors from overheating present an issue when motion compensating for a fast moving heart. The RAVEN also only shut down due to exceeding current limits in certain configurations as end effector placement in the workspace affects torque requirements. And lastly, with the goal of uncovering considerations necessary to move tissue tracking and motion compensation to real-time implementation in clinics, we found that the main hurdle in our methods should not be an impenetrable barrier for this technology. Image processing time with an outdated CPU kept our tests operating at one-eighth real-time but current generation processors are likely to enable real-time implementations of our methods, or will in the near future.

The following section details extensions of this project to further knowledge while working towards clinical use.

6.1 Future Work

More testing is required to further explore optimum ROI size and number of control points, robot current constraints, decreasing computation time, and tracking realistic tissue surfaces.

The utilized test procedure to find the maximum number of control points a given ROI size could successfully track returned surprising results that are not fully understood. A more detailed analysis of additional testing could quantify relative computation costs, measured with units of time, for manipulating the number of tracked pixels and control points. It seems reasonable that fitting a model to these results could determine the most efficient ROI size for a desired level of tracking detail.

Further analysis of current limitations for the specific motors used by the RAVEN and other surgical robots in addition to the effects of workspace location on current requirements could enable a more robust motion compensating scheme. An accurate model of safe current levels as a function of time for each joint combined with knowledge of likely future current demands given the present position of end effectors could enable an intelligent current allocator which protects motors with minimal interference to control.

A few implementations of the presented methods in the literature have utilized GPUs to decrease computation time. We would also like to develop a GPU optimized version of this work to reevaluate progress towards real-time tracking and find additional considerations when operating much closer to real-time.

And lastly, testing was limited in that it only simulated rigid body motion while real heart surfaces also deform. The TPS was chosen as the surface modeling algorithm specifically for its support of deformable surface tracking, leading to potential future tests with a deformable surface without any program alterations. This would reveal performance of the TPS modeling real tissue and any resulting computation effects.

BIBLIOGRAPHY

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3951 LNCS:404–417, 2006.

[2] Özkan Bebek and M Cenk Çavusoglu. Intelligent Control Algorithms for Robotic-Assisted Beating Heart Surgery. *IEEE Transactions on Robotics*, 23(3):468–480, 2007.

[3] G.S. Guthart and J.K. Salisbury. The Intuitive telesurgery system: overview and application. *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, 1(April):618–621, 2000.

[4] Blake Hannaford, Jacob Rosen, Diana W. Friedman, Hawkeye King, Phillip Roan, Lei Cheng, Daniel Glozman, Ji Ma, Sina Nia Kosari, and Lee White. Raven-II: An open platform for surgical robotics research. *IEEE Transactions on Biomedical Engineering*, 60(4):954–959, 2013.

[5] J.W. Hill, P.S. Green, J.F. Jensen, Y. Gorfu, and a.S. Shah. Telepresence surgery demonstration system. *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 2302–2307, 1994.

[6] W W Lau, N A Ramey, J J Corso, N V Thakor, and G D Hager. Stereo-based endoseopic tracking of cardiac surface deformation. *Medical Image Computing and Computer-Assisted Intervention - Miccai 2004, Pt 2, Proceedings*, 3217:494–501, 2004.

[7] Mitchell J H Lum, Diana C W Friedman, Hawkeye H I King, Regina Donlin, Ganesh Sankaranarayanan, Timoty J. Broderick, Mika N. Sinanan, Jacob Rosen, and Blake Hannaford. Teleoperation of a surgical robot via airborne wireless radio and transatlantic internet links. *Springer Tracts in Advanced Robotics*, 42:305–314, 2008.

[8] E Malis and S Benhimane. Homography-based 2D Visual Tracking and Servoing. *The International Journal of Robotics Research*, 26(7):661–676, 2007.

[9] Peter Mountney and Guang-zhong Yang. Soft Tissue Tracking for Minimally Invasive Surgery :. *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, 11 VN - r(Pt 2):364–372, 2008.

[10] Tobias Ortmaier, Martin Gröger, Dieter H. Boehm, Volkmar Falk, and Gerd Hirzinger. Motion estimation in beating heart surgery. *IEEE Transactions on Biomedical Engineering*, 52(10):1729–1740, 2005.

[11] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Mg. ROS: an open-source Robot Operating System. *Icra*, 3:5, 2009.

[12] Rogério Richa, Antônio P L Bó, and Philippe Poignet. Towards robust 3D visual tracking for motion compensation in beating heart surgery. *Medical Image Analysis*, 15(3):302–315, 2011.

[13] Rogerio Richa, Philippe Poignet, and Chao Liu. Three-dimensional Motion Tracking for Beating Heart Surgery Using a Thin-plate Spline Deformable Model. *The International Journal of Robotics Research*, 29(2-3):218–230, 2010.

[14] Rogerio Richa, Raphael Sznitman, Russell Taylor, and Gregory Hager. Visual tracking using the sum of conditional variance. *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (2):2953–2958, 2011.

[15] Mickaël Sauvée, Philippe Poignet, Jean Triboulet, Etienne Dombre, Ezio Malis, and Roland Demaria. 3D Heart Motion Estimation Using Endoscopic Monocular Vision System. *IFAC Proceedings Volumes*, 39(18):141–146, 2006.

[16] Danail Stoyanov, Ara Darzi, and Guang Zhong Yang. A practical approach towards accurate dense 3D depth recovery for robotic laparoscopic surgery. *Computer aided surgery : official journal of the International Society for Computer Aided Surgery*, 10(July):199–208, 2005.

[17] Danail Stoyanov, George P. Mylonas, Fani Deligianni, Ara Darzi, and Guang Zhong Yang. Soft-tissue motion tracking and structure estimation for robotic assisted MIS procedures. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3750 LNCS:139–146, 2005.

[18] E. Erdem Tuna, Jamshid H. Karimov, Taoming Liu, Özkan Bebek, Kiyotaka Fukamachi, and M. Cenk Çavuolu. Towards active tracking of beating heart motion in the presence of arrhythmia for robotic assisted beating heart surgery. *PLoS ONE*, 9(7):3–10, 2014.

[19] Menglong Ye, Stamatia Giannarou, Alexander Meining, and Guang Zhong Yang. Online tracking and retargeting with applications to optical biopsy in gastrointestinal endoscopic examinations. *Medical Image Analysis*, 30:144–157, 2016.