On the Development of a Heart Motion Compensation System on the da Vinci Research Kit for Minimally Invasive Surgery on the Beating Heart

by

Angelica Ruszkowski

BASc, University of Waterloo, 2013

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Applied Science

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Electrical and Computer Engineering)

The University of British Columbia

(Vancouver)

August 2015

© Angelica Ruszkowski, 2015
Abstract

This thesis describes the development of a heart motion compensation system on the da Vinci Research Kit for coronary artery bypass surgery. With this teleoperation robotic platform, minimally invasive surgery on a beating heart could be performed on an already clinically prevalent system. Semi-automation of the slave manipulators of the robot is introduced as they track the surface of the beating heart. The surgeons regular teleoperation commands are superimposed on the automated trajectory. To achieve a virtually stabilized environment, a novel concept of maintaining the camera fixed relative to the heart target is proposed.

The preliminary research question is whether the robot is capable of tracking the highly dynamic heart motion. System identification was performed on the seven degree of freedom da Vinci slave manipulators, and an open loop controller was developed. The controller is based on spectral line decomposition and the assumption of a periodic trajectory. It successfully commanded the slave manipulators to track an actual three dimensional heart trajectory with submillimetre error. Experiments were conducted with expert robotic users to evaluate surgeons’ ability to perform tasks on a moving target emulating the beating heart, with very promising outcomes. The number of missed targets decreased from 37% to 13% when compensation was enabled, the number of hit targets increased from 26% to 41%, and completion time decreased.

A second generation system was developed which includes real time motion measurement commanding the robot. Results from user studies with expert surgeons performing bimanual suturing on moving targets with the new system support the motion compensation. They also show the significance of motion measurement errors.
As an added safety, a virtual fixture was implemented to protect the heart from accidental collisions with the instrument tips. User studies were conducted to validate the efficacy of the fixture.

To expedite controller development, an interface was developed between Matlab Simulink and the C++ code that runs the da Vinci Research Kit. This allows on-the-fly testing of controllers which could be designed and developed in the convenient Simulink environment.

Future work will be include closed loop control, improved experiment design, and the incorporation of electrocardiogram signals.
Preface

I was the lead investigator for the work described herein. My supervisor, Dr. Salcudean, presented me with the project of heart motion compensation on the da Vinci Research Kit. The design, development, implementation and user studies on the proof of concept heart motion compensation system were all conducted in the Robotics and Controls Laboratory (RCL) in the Electrical and Computer Engineering Department at the University of British Columbia, Point Grey Campus. I was granted a visiting student position at the Collaborative Haptics and Robotics in Medicine (CHARM) Laboratory in the Mechanical Engineering Department at Stanford University for the last quarter of 2014. At Stanford, I was the primary researcher on two collaborative projects, one infrastructure-based and one research-based. Publications resulted from both of these, as described below. After Stanford, I returned to RCL where I continued to be the lead investigator in the completion of system improvements including the development of more advanced control techniques and more clinically relevant user studies.

A version of Section 5.3 was presented as a poster at the workshop discussing Shared Frameworks for Medical Robotics Research at the International Conference for Robotics and Automation (ICRA) 2015 in Seattle, Washington [1]. As associated abstract was submitted and is to be included in the workshop proceedings [Ruszkowski A., Quek, Z. F., Okamura, A. M., and Salcudean, S. Simulink® to C++ Interface for Controller Development on the da Vinci® Research Kit (dVRK). ICRA 2015 Workshop: Shared Frameworks for Medical Robotics Research]).

studies. In Robotics and Automation (ICRA), 2015 IEEE International Conference on (pp. 4432-4439). IEEE] has been partitioned into various sections of this thesis. The manuscript was edited for its inclusion herein to maintain the logical flow of the thesis. Controller development details from the publication are included in Section 5.1, Section 5.4 and Section 5.5. The user studies, results and conclusions described in the publication are found in Chapter 6. The system described in this publication was also demonstrated as an entry at the first Surgical Robot Challenge at the 2015 Hamlyn Symposium for Medical Robotics in London, UK.

A version of Chapter 7 has been presented as a poster with an associated abstract at the 2015 Hamlyn Symposium for Medical Robotics [Ruszkowski A., Quek, Z. F., Okamura, A. M., and Salcudean, S. Dynamic Non-Continuous Virtual Fixtures for Operations on a Beating Heart Using the da Vinci® Research Kit. 2015 Hamlyn Symposium on Medical Robotics (pp. 45-46)]. Quek Z.F. contributed to the early stages of concept formation.

For all of the above, as lead investigator for this work I was responsible for concept development, software implementation, user study design and execution, data collection and analysis, and manuscript composition. Dr. Okamura co-supervised the research we conducted at Stanford, offering her expertise in haptics for concept formation. Dr. Salcudean was the supervisory author on this entire project and was integral throughout the project in concept formation and development, manuscript edits, and most importantly technical guidance.
Table of Contents

Abstract ................................................................. ii
Preface ................................................................. iv
Table of Contents .................................................... vi
List of Tables ........................................................... x
List of Figures ........................................................... xi
Glossary ................................................................. xv
Acknowledgments ....................................................... xvii

1 Introduction .......................................................... 1
   1.1 Coronary Artery Bypass Grafting Surgery .......................... 1
   1.2 The da Vinci Research Kit ............................................. 2
   1.3 Available Solutions ................................................... 3
   1.4 Project Objectives ................................................... 3
   1.5 Thesis Overview ..................................................... 4

2 Literature Review .................................................... 6
   2.1 Coronary Artery Bypass Grafting .................................... 6
       2.1.1 Background .................................................... 6
       2.1.2 Procedure Description ....................................... 7
       2.1.3 Advances in CABG ............................................. 8
   2.2 Previous Work ..................................................... 10
       2.2.1 Motion Measurement ........................................ 10
8.4.2 Discussion .......................................................... 103
8.5 Conclusions and Future Work ........................................ 104

9 Conclusion and Recommendations ...................................... 107
  9.1 Conclusions .......................................................... 107
  9.2 Contributions ....................................................... 109
  9.3 Recommendations .................................................. 110

Bibliography .............................................................. 113

A Running an Experiment ................................................. 120

B Software Overview ...................................................... 124
  B.1 SVN ................................................................. 124
      B.1.1 HeartTrajectoryCode ......................................... 125
      B.1.2 LogDataAnalysis ............................................. 125
      B.1.3 NDICalibResults .............................................. 125
      B.1.4 NDICalibration ............................................... 125
      B.1.5 NDIOAPI_svn ................................................. 125
  B.2 GIT ................................................................. 125
List of Tables

Table 4.1  Control loop frequencies ......................................................... 32
Table 5.1  Experimentally Determined Transfer Function Parameters for PSM Joints 40
Table 5.2  Timing analysis for Simulink joint-level PID (PIDθ) and Cartesian (PIDx) controllers ............................................................. 49
Table 5.3  Ranked harmonics of the heart motion ........................................ 51
Table 5.4  Tracking Errors using Spectral Line Controller .............................. 53
Table 8.1  Maximum Tracking Errors for Closed Loop Heart Motion Compensation System ................................................................. 89
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>A graphic depicting the Coronary Artery Bypass Grafting (CABG) procedure</td>
<td>7</td>
</tr>
<tr>
<td>2.2</td>
<td>A classic end-to-side anastomosis</td>
<td>8</td>
</tr>
<tr>
<td>2.3</td>
<td>The highly invasive sternotomy</td>
<td>9</td>
</tr>
<tr>
<td>2.4</td>
<td>The fulcrum effect in the context of hand instruments of laparoscopic surgery</td>
<td>10</td>
</tr>
<tr>
<td>2.5</td>
<td>The Octopus by Medtronic</td>
<td>12</td>
</tr>
<tr>
<td>2.6</td>
<td>The Cardiolock device</td>
<td>14</td>
</tr>
<tr>
<td>2.7</td>
<td>The GyroLock device</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>Direction of axes of heart motion</td>
<td>21</td>
</tr>
<tr>
<td>3.2</td>
<td>Translation components of original versus rotated heart motion trajectories</td>
<td>22</td>
</tr>
<tr>
<td>3.3</td>
<td>Comparison of translational components of used heart trajectory.</td>
<td>22</td>
</tr>
<tr>
<td>3.4</td>
<td>3D view of used heart trajectory</td>
<td>23</td>
</tr>
<tr>
<td>3.5</td>
<td>Kinematics of heart trajectory</td>
<td>25</td>
</tr>
<tr>
<td>3.6</td>
<td>Spectral analysis of heart trajectory</td>
<td>25</td>
</tr>
<tr>
<td>4.1</td>
<td>The dVRK System at UBC</td>
<td>27</td>
</tr>
<tr>
<td>4.2</td>
<td>The Master Console of the da Vinci Classic</td>
<td>28</td>
</tr>
<tr>
<td>4.3</td>
<td>The Patient Side Manipulators of the da Vinci Classic</td>
<td>28</td>
</tr>
<tr>
<td>4.4</td>
<td>Controller boxes for the dVRK</td>
<td>29</td>
</tr>
<tr>
<td>4.5</td>
<td>Degrees of motion of a PSM</td>
<td>32</td>
</tr>
<tr>
<td>4.6</td>
<td>Remote center of motion on a PSM</td>
<td>33</td>
</tr>
<tr>
<td>4.7</td>
<td>DH Frames for the PSM and attached (Large Needle Driver) instrument</td>
<td>34</td>
</tr>
<tr>
<td>5.1</td>
<td>Block diagram representation of the PSM motor plant</td>
<td>37</td>
</tr>
</tbody>
</table>
Glossary

AOB  Active Observer

API  Application Programmer’s Interface

ARMC  Active Relative Motion Cancelling

bpm  Beats per Minute

CABG  Coronary Artery Bypass Grafting

CPB  Cardiopulmonary Bypass

DH  Denavit-Hartenberg

DOF  Degree of Freedom

dVRK  da Vinci Research Kit

ECABG  Endoscopic Coronary Artery Bypass Grafting

ECG  Electrocardiogram

ECM  Endoscopic Camera Manipulator

EKF  Extended Kalman Filter

FDA  Food and Drug Association

FIFO  First In First Out

FPGA  Field Programmable Gate Array

fps  Frames Per Second
HIP  Haptic Interface Point
ICRA  International Conference for Robotics and Automation
MCI  Motion Compensation Instrument
MPC  Model Predictive Control
MRI  Magnetic Resonance Image
MTM  Master Tool Manipulator
OPCABG  Off-Pump Coronary Artery Bypass Grafting
PCA  Principal Component Analysis
PCI  Percutaneous Coronary Intervention
PD  Proportional-Derivative
PID  Proportional-Integral-Derivative
PSM  Patient Side Manipulator
QLA  Quad Linear Amplifier
RCM  Remote Center of Motion
RNEA  Recursive Newton Euler Algorithm
RHMPC  Receding Horizon Model Predictive Control
SISO  Single Input Single Output
SVD  Singular Value Decomposition
TPS  Thin-Plate Spline
US  Ultrasound
Acknowledgments

First, I would like to express my deepest gratitude to my supervisor Dr. Tim Salcudean, for giving me the opportunity of working on one of the most fascinating projects on the absolute coolest toy in perhaps the most beautiful city in the world. Dr. Salcudean’s brilliant technical guidance and support was integral to the successful completion of this project. Thank you for all the advice, assistance with formulating battle plans, letting me run free and then helping me when I got completely stuck. Also, thank you to both Dr. Salcudean and my temporary co-supervisor, Dr. Okamura, for the invaluable experience of being a Visiting Student at Stanford University.

I would also like to share my sincere appreciation for our clinical collaborators, namely Dr. Sam Lichtenstein, Dr. Richard Cook, Dr. Larry Goldenberg, Dr. Peter Black, and Dr. Chris Nguan. It is incredible to have such support on the clinical side, and enthralling to have had expert robotic surgeons testing (and liking!) my system.

The financial support from NSERC CGS-M, NSERC MSFSS and Microsoft Research Women’s Scholarship Program, as well as from CFI, the C.A. Laszlo Chair, and NSF Grant #1227406, is all very gratefully acknowledged.

I would like to recognize and express my appreciation to Omid Mohareri and Zhan Fan. These two PhDs were the da Vinci robot masters at UBC and Stanford, respectively, and I owe them many many thanks. On the topic of the robot, I must also recognize Peter Kazanzides and Anton Deguet from Johns Hopkins University, for all your amazing work and support on the dVRK. It was especially appreciated all the times I managed to break the robot. The incredible contribution of Simon diMaio must also be acknowledged; from lending me two prototype camera systems to helping me with robot maintenance and technical
questions about the robot, thank you for everything.

I have to thank all my Vancouver/VOC friends and RCL labmates, particularly the foosball squad, the Spartacus crew, and my late night/weekend lab buddies. You’ve made my experience at UBC an amazing one. A very special thanks to Caitlin Schneider and Julio Lobo. These two helped me maintain my sanity and get through my first user studies with the surgeons. Caitlin, also thanks for pulling me into the VOC before I even got here and for all the outdoors advice and suggestions. I’m going to miss this playground around Vancouver immensely.

Also, here’s to the UBC Figure Skating Club whose pilot program ran my last spring term here! Best of luck in the first official years of the club!

My deepest love and gratefulness is felt for my parents Barbara and Kazimierz Ruszkowski, my sister Dominika and brother Sebastian. Thank you for all the encouragement, love, Skype calls and support when I was so far from home. To Mama and Tata, I cannot really express my gratitude for everything you have been through, everything you’ve sacrificed, everything you’ve done to get me to where I am. You are my heroes and role models. Bardzo bardzo was kocham, dwa razy bardziej.

Finally, to my dearest Brian, thank you for your extraordinary love and support. Despite the long distance, you helped me get through these past two years. From helping me understand controls concepts, to patiently listening to my rants about the lab; from being a neverending source of comfort when things got tough to listening to all my excited tales of my adventures here – you are always there for me. Thank you, my Love, it’s been an intense two years, I would have lost my sanity long ago if not for you.
Chapter 1

Introduction

1.1 Coronary Artery Bypass Grafting Surgery

Cardiovascular diseases are the leading cause of death in the world, accounting for nearly one third of all global deaths in 2008 [2]. There is also a significant economic impact sustained from cardiovascular disease. In 2009, a total of $312.6 billion direct and indirect costs were incurred in the United States alone [3]. In advanced coronary artery diseases, where a blockage develops in one of the coronary arteries, the most common surgical revascularization procedure is called Coronary Artery Bypass Grafting (CABG) Surgery.

As the name indicates, the objective of the CABG procedure is to bypass a blocked coronary artery. This is accomplished by harvesting an artery from elsewhere in the patient and then attaching (or grafting) it to the surface of the heart to bypass the blockage. The vessels involved are typically in the order of 2 mm or less in diameter, and so the procedure requires very fine precision. To make this task feasible for the surgeon, the heart is normally stopped at the patient is put on a Cardiopulmonary Bypass (CPB) machine. It has been documented, however, that the use of the CPB incurs significant possibility of complications including cognitive loss and increased hospitalization time and costs [4]. Moreover, a sternotomy is performed to access the heart. This involves a large incision and cracking the sternum – a highly painful procedure with long recovery times.

The achievement of minimally invasive beating heart surgery would be greatly advantageous for the patient, eliminating the need for both the sternotomy and the use of the CPB
machine. The concept of heartbeat synchronization, or motion cancellation, is promising for achieving this goal. By devising a system which can measure the heart motion and track it, the instruments could be commanded to keep a fixed distance from the beating heart surface. The surgeon’s motions would thus be relative to the tissue, and ideally the high precision surgical manoeuvres could be preserved.

1.2 The da Vinci Research Kit

The motion tracking component of a heart motion compensation system lends itself perfectly to a robotic solution. Programming a robot to automatically compensate for the heart motion would provide a feasible solution to the challenge of minimally invasive beating heart surgery, assuming the robot was capable of following the highly dynamic motion of the heart.

The da Vinci® Surgical System from Intuitive Surgical Inc. in Sunnyvale, California is already a clinically prevalent surgical robot. Thousands are presently in use in hospitals, with over half a million robotic procedures having being performed in 2014 alone. The da Vinci is a teleoperation robot; it comprises of two major components, the master and slave. The surgeon sits at the Master Console, typically a few feet away from the patient, and operates the Master Tool Manipulator (MTM)s. The surgeon’s motions are read by the robot and translated to the patient by way of the Patient Side Manipulator (PSM)s. The robot performs filtering to remove the surgeon’s hand tremors, motion scaling to permit high precision motions on the slave side, and kinematic transformations that align the surgeon’s physical hand motions with the motions of the instruments in the camera feed. The instruments are articulated and offer high degrees of dexterity in the minimally invasive environment. The vision system provides the da Vinci operator with a 3D view of the surgical field.

In collaboration with Intuitive Surgical Inc., Johns Hopkins University and Worcester Polytechnic Institute have developed the da Vinci Research Kit (dVRK). The dVRK is a mechatronics telerobotic surgical research platform comprising of mechanical components from the first generation da Vinci Surgical System (the da Vinci Classic), as well as electronic hardware and software offering researchers access to all levels of control of the robot. Using this platform, researchers are able to experiment with new control and automation
schemes.

1.3 Available Solutions

Since the benefits of minimally invasive beating heart surgery are great, much work has gone into trying to accomplish this challenge. Clinically, Endoscopic Coronary Artery Bypass Grafting (ECABG) is attempted using mechanical stabilizers such as the Octopus® tissue stabilizers. Instead of actively compensating for the heart motion, these devices suppress the heart motion by suctioning themselves down to the heart surface and restricting the movement of tissue. Studies have shown that the residual motion of these devices is still too large (about 1.5–2.4 mm), and safety and graft patency remain major concerns [5],[6],[7]. Active mechanical stabilizers have been developed to try to alleviate these concerns [8]. Epicardial crawling robots offer an interesting solution to compensating for the heart motion [9]. A number of custom heart motion compensation systems have also been developed [10],[11],[12],[13]. For the aforementioned solutions, however, clinically feasibility and the ability to acquire approval from the Food and Drug Association (FDA) for clinical use remain major challenges.

1.4 Project Objectives

The goal of this thesis is to perform a feasibility study on achieving minimally invasive beating heart surgery using the dVRK as a platform. This is a novel use of the da Vinci and the project will include

- a detailed evaluation of the robotic platform to validate its capability to track the complex, highly dynamic heart motion with sufficient accuracy.
- controlling the PSMs of the da Vinci to automatically track the heart motion. This will keep the instruments at a fixed distance from the beating heart surface.
- the development of specialized controllers to achieve submillimetre tracking errors.
- achieving visual stabilization with a small stereo endoscopic camera which will also track the heart motion, to provide the robot operator the illusion of working in a virtually stabilized field.
• when working on a moving target, *emulating performing surgery on a stopped heart* by superimposing the surgeon’s teleoperation commands on the automated trajectories.

• *incorporating motion measurement for real time control* of the da Vinci PSM based on the measured position of the heart surface

• completing *user studies* to evaluate the participants’ abilities to use the proposed heart motion compensation system to accomplish tasks on a moving target.

The main advantage of this work is its clinical feasibility; it is being developed on the da Vinci system which is already clinically prevalent. With an existing widespread presence in clinical practice, the da Vinci is poised to be a platform through which minimally invasive surgery on the beating heart could have a significant impact on cardiovascular procedures.

### 1.5 Thesis Overview

The structure of this thesis is organized as follows:

• **Chapter 1 Introduction**: Discusses the motivation and objectives of this thesis.

• **Chapter 2 Literature Review**: Provides a background for the targeted clinical procedure, discusses previously proposed solutions for motion compensation, and describes the technical challenges of minimally invasive surgery on the beating heart.

• **Chapter 3 Heart Motion**: Offers a description of the heart trajectory used in this study, including data processing and kinematics.

• **Chapter 4 Platform Used / The System**: Provides a detailed description of the robotic platform, including its configuration, kinematic and dynamics, and control architecture.

• **Chapter 5 Controller Design**: Explains the design and development of the various controllers used to achieve high accuracy heart motion tracking.

• **Chapter 6 Initial User Studies**: Discusses the design and implementation of the initial proof of concept heart motion compensation system and corresponding user studies.

• **Chapter 7 Virtual Fixtures**: Describes the development and evaluation of active constraints that aim to improve safety of the heart motion compensation system.
• Chapter 8 Incorporating Motion Measurement: Details the extension of the system to incorporate real time motion measurement, moving towards clinical feasibility.

• Chapter 9 Conclusions and Recommendations: Concludes the thesis with an overview of this project. Also suggests recommendations for future improvements and developments.
Chapter 2

Literature Review

This chapter presents a review of the literature relevant to this project. The first section describes the target application, CABG, in more detail. Next, pertinent previous work towards heart motion compensation systems are discussed. The last two sections explore the control systems that have been previously developed for cardiac applications.

2.1 Coronary Artery Bypass Grafting

2.1.1 Background

Coronary heart disease is a disease in which blood flow to the heart is reduced due to a plaque build up inside the coronary arteries. The arteries are very narrow blood vessels that carry nutrients and oxygen to the heart. When the occlusion in the arteries builds, the flow of oxygen is reduced and blood clots may form, potentially completely blocking blood flow. Angina and cardiac arrests may both be caused by such blockages \[14\].

If the coronary artery disease is not too advanced, a nonsurgical procedure called Percutaneous Coronary Intervention (PCI) (formerly known as angioplasty with a stent) may be performed. PCI uses a catheter to insert a stent (a small mesh balloon that supports the inner artery wall) to open up blood vessels \[15\]. In some cases PCI is insufficient to improve blood flow through the arteries and the more invasive CABG procedure must be performed.
CABG is the most common open-heart surgery, with upwards of 400,000 patients in the 2009 in the United States [16]. It is an highly effective treatment with typically excellent results. Two large, independent studies (FREEDOM and SYNTAX) were recently performed and both concluded that CABG should be the preferred revascularization strategy for treating coronary artery disease [17],[18]. This would suggest a rising number of future cases and greater potential impact for a significant improvement in the procedure.

2.1.2 Procedure Description

The CABG procedure aims to bypass a blockage formed from plaque build up inside the coronary artery. This is accomplished by harvesting an artery from elsewhere in the patient and attaching, or grafting, it to the surface of the heart to bypass the blockage. A diagram is seen in Figure 2.1. The saphenous vein from the leg and the left internal thoracic artery are most common bypass grafts, with the latter being preferred since they tend to remain open longer than venous grafts [21]. One end of the graft is attached to the aorta, and the other to the coronary artery distal to the blockage by way of an end-to-side anastomosis (Figure 2.2).

The vessels being anastomosed in the CABG procedure are in the order of 2 mm in diameter. Thus great precision is required to complete the suture task. To accomplish this,

**Figure 2.1:** A graphic depicting the CABG procedure. Venous or arterial grafts are made to bypass blockages in the coronary artery [19]
the heart is usually stopped and the patient is put on a CPB machine. The CPB provides physiological support with an extracorporeal circuit, resulting in a motionless, bloodless surgical field. The circuit drains blood from the heart and lungs via venous cannulation and tubing, oxygenates it, and returns the re-oxygenated blood to the cannulated arterial system in a constant flow using a mechanical pump and artificial lung [22]. Use of the CPB results in a series of secondary damaging effects, including anemia, red blood cell aggregation, gaseous and particulate emboli, hemolysis (red blood cell damage), and localized ischemia [4].

Since CABG is traditionally an open procedure, a sternotomy is performed in order to access the heart. This procedure involves a several inch long incision down the center of the chest and cracking the sternum open, seen in Figure 2.3. This is a highly painful and invasive procedure, often requiring months of recovery.

2.1.3 Advances in CABG

The invasiveness of traditional CABG surgery serves as motivation to investigate two major changes in the procedure. The first major change eliminates the need for the CPB
Term Off-Pump Coronary Artery Bypass Grafting (OPCABG), this means performing the surgery on the beating heart. The motion of the heart is a highly complex, location-dependent, six dimensional motion. Secondly, to eliminate the need for the invasive sternotomy, there have been efforts to proliferate minimally invasive, or ECABG. For such a procedure, a number of small incisions would be made between the ribs, and the heart would be accessed using long shafted instruments in a laparoscopic setting. With classic laparoscopy this would be exceedingly challenging due to the reduced mobility, visibility and ergonomics, as well as the introduction of the fulcrum effect. In the context of laparoscopic surgery, seen in Figure 2.4, the fulcrum effect results in magnified hand motions (including hand tremors) and the direction of the tool tip is inverted from the direction of hand motion. Despite these challenges, the potential advantages of minimally invasive surgery are compelling, including reduced patient trauma and lowered surgery costs.

Robotic techniques have proven very successful for reducing the complexity for surgeons in minimally invasive surgery, particularly in urological procedures. The da Vinci®Surgical System, specifically, is a teleoperation robot already in clinical use. The da Vinci has the ability to filter out hand tremors, scale the surgeon’s motions, provide increased dexterity in the workspace through wristed, articulated instruments, and intelligent controls which align the surgeon’s physical motions to the instrument motion in the camera feed. A study in Beijing attempted to perform the entire CABG procedure minimally invasively on the beating heart with the da Vinci, but there was a substantial learning curve for even the expert robotic surgeon [7]. As a compromise, a cardiothoracic surgeon at Vancouver General
Figure 2.4: The fulcrum effect in the context of hand instruments of laparoscopic surgery [24].

Hospital performs the majority of the CABG surgery robotically, however for the actual anastomosis task, a minor sternotomy must still be performed and the patient is still put on the CPB machine. Should there be a system capable of performing the anastomosis on the beating heart, off-pump ECABG could be achieved.

2.2 Previous Work

A complete active heart motion compensation system would consist of two major components: a motion measurement component and a motion tracking component. The former would use a sensor technology that is able to read the position of the heart surface tissue in space and relay this information to some controller. The controller would use this data to command the device responsible for physically tracking the heart surface. Each of these components involves considerable complexity. The major focus of this thesis is on the motion tracking component, and as such this was more thoroughly investigated in the literature. Some work on motion measurement will still be discussed, though briefly.

2.2.1 Motion Measurement

Some of the earliest attempts at measuring the motion of the heart surface involve vision systems (cameras) and embedded markers positioned at strategic locations (e.g. bifurcation points of the coronary arterial tree) on the heart surface [25]. [26] was one of the
first groups to perform feature-based soft tissue motion tracking of the heart surface. Soft tissue tracking presents several challenges, namely the tissue is deformable, there are specular reflectances from the bright camera lighting, occlusions from instruments or blood, as well as a general lack of distinctive features. The approach taken by Stoyanov et al. [26] to measure the movement of the tissue is to track 3D points in stereo images from a pre-calibrated stereo laparoscope using stereo-temporal constraints and an iterative registration algorithm. Their salient landmark selection employs maximally stable extremal regions based on thresholding the image intensity, as well as traditional gradient-based image features to create connected components. Temporal motion tracking is achieved using a Lucas-Kanade tracker and registration.

An efficient algorithm for 3D tracking of the heart surface based on a Thin-Plate Spline (TPS) deformable model and an illumination compensation algorithm was proposed by Richa et al. [27]. A vision-based technique, the TPS warping defines a mapping between each pixel to the current image of the surface, using a parametrization of projective depths of the tracked surface to represent projective deformations. Illumination parametrization is also incorporated. An efficient second-order minimization provides a fast and large convergence basin. Initial results were very promising, but there is a trade off between the number of control points and computation costs. With a lower number of control points, occlusions become more of an issue.

Kurz et al. discuss a multiple target tracking technique in [28]. To achieve precise tracking of the heart surface, 24 identical artificial markers are placed on the heart. To determine which measurement came from which marker, the multiple target tracking method is based on a symmetric kernel equation transformation. They showed the algorithm capable of reliably tracking 24 unlabeled targets in real time. Clinical issues such as instrument or blood occlusion of the markers, as well as placement of the markers, remain practical concerns.

### 2.2.2 Motion Tracking

This section will provide an overview of existing/attempted compensation systems (both passive and active approaches will be discussed). These compensation systems are typically mechatronics systems consisting of mechanical hardware, electronics and software for control. There will also be a discussion of more advanced control techniques that have been investigated.
2.2.2.1 Passive Compensation Systems

Mechanical surgical stabilizers are a type of device in clinical use which could be described as passive compensation systems. The Octopus®, developed by Medtronic [29], is a common example, seen in Figure 2.5. A similar device was developed by the University of British Columbia [30]. These stabilizers use suction or vacuum pressure to clamp down on the heart tissue surrounding the surgical site, intending to stabilize the local tissue. There are risks of damaging the tissue if too much vacuum/suction force is used, and studies have shown the residual motion (about 1.5–2.4 mm) is too great to achieve the accuracy required for highly effective anastomoses [5],[6].

The HeartLander robot, developed by Riviere et al. [9] is a very interesting concept of an epicardial crawling robot which eliminates the need for active motion compensation. A small robot with a magnetic tracking sensor traverses the surface of the heart using inchworm-like locomotion. By modelling the physiological motion as a time-varying Fourier series using an Extended Kalman Filter (EKF) framework, more accurate position estimation as well as synchronization of motion to the physiological cycles (both respiratory and cardiac) can be achieved. Such a device is not directly applicable to CABG surgery, but is useful for modelling heart motion and administering of treatments locally.

2.2.2.2 Active Compensation Systems

An active compensation system in this context is a system which addresses mechanical stabilization, image stabilization, and shared control in order to perform beating heart surgery. One of the earliest attempts at building such a system was by Trejos et al. [10].
The proposed system involved a hand/tool support which moved the surgeon’s hands to track the motion of the heart. The device consisted of a specifically designed 3 Degree of Freedom (DOF) mechanism with prismatic joints, a custom made mechanical sensor, and a suitable control strategy comprising of Proportional-Integral-Derivative (PID) control and computed torque control. A simulated suture task was attempted in a user study and it was demonstrated that moving the hand in synchrony with the task space achieved significant improvement in accuracy over the uncompensated version. An increase in task completion time was also observed.

Nakamura et al. [12] introduced the concept of heartbeat synchronization for minimally invasive cardiac surgery by building a custom teleoperation robot. A Phantom Desktop haptic device (by SensAble Technologies Inc.) was adopted as the master device, and a prototype 4 DOF arm was used for the slave. The visual synchronization was done electronically, processing the camera image by cutting and translating it so a laser-indicated reference point always remained in the same position. Motion measurement was achieved using a 955 Frames Per Second (fps) high speed camera to measure vibration of the laser pointer. Experimental results were very promising, unfortunately this cutting edge system was not developed further by this group.

Instead of making a teleoperation robot, Yuen et al. [11] consider an actuated handheld 1 DOF Motion Compensation Instrument (MCI). This work investigates the approximation of tissue motion as a 1D motion model, specifically the motion of the mitral valve annulus. 3D Ultrasound (US) images are utilized to perform the motion measurement, however these introduce substantial temporal delays. A predictive filter, the EKF, exploits the quasiperiodicity in the cardiac motion and is used to compensate for these delays. Using a feedforward trajectory of the cardiac target, user study results demonstrated that the 1D motion synchronization MCI allowed users to operate with increased dexterity and reduced forces on the simulated heart surface.

Another approach that builds upon the passive cardiac stabilizers discussed previously is one which introduces active compensation to eliminate the residual motion in the passive stabilizers. In [8] two such active stabilizers are presented: the Cardiolock and the GyroLock systems. Cardiolock introduces a piezo-actuator in the stabilizer structure, while GyroLock works on the principle of the generation of gyroscopic torque. The cardiac stabilizers are inserted on a long stabilizer rod, and are tasked with stabilizing the heart surface.
in 2 DOF, those perpendicular to the beam. In the Cardiolock, shown in Figure 2.6, high
dynamic actuators, piezoelectric actuators on a slide-crank mechanism, are introduced be-
tween the stabilizer rod and its base. This mechanism controls the configuration of the
compliant mechanism to cancel any stabilizer tip displacement, which is observed using
the endoscopic camera. A custom end-effector is added to the instrument, hosting a 6 DOF
force sensor and visual marker. The closed loop controller accomplished submillimetre
standard deviation of the residual displacement. A very enticing solution, clinical feasi-
bility will be dependent on the quality and framerate of the endoscopic camera feed. The
GyroLock, in Figure 2.7, is designed to plug onto commercial passive stabilizers without
modification. Operating based on inertial effects, the GyroLock employs the principle of
Control Moment Gyroscope actuation, utilizing acceleration measurements taken from a
sensor embedded at the stabilizer tip. By controlling the inclination of a gyroscope spin-
ning at a constant high speed, an alternating torque can be generated which allows for the compensation of periodic motion. An adaptive algorithm dedicated to harmonic disturbance rejection is used for the control law. Experimental results showed the reduction of displacement is sufficient for heart stabilization. Another highly interesting solution, however the authors did not discuss the ease of modification of the commercially available stabilizer, nor a medical expert’s perspectives on whether the large device will be welcomed in the already cramped operating theatre.

Finally, a system named the MicroSurge is being developed by the DLR Institute of Robotics and Mechatronics in Germany [31]. The system is designed to achieve a high degree of versatility in minimally invasive surgery, performing bimanual endoscopic telesurgery with force feedback. Literature describing the design of the system – arms, instruments, master workstation, planning, etc. – is available, however experimental results for specific applications have not yet been disclosed. With an ultimate ambition to perform robot-supported surgery on the beating heart, this is a group to watch for future innovations.

While very creative and often promising, the majority of these systems cannot be clinically feasible without considering the process of obtaining FDA approval, which is both highly challenging and time consuming.

2.2.2.3 Control Systems

A number of groups focused more intently on the intelligent control algorithms for robotic-assisted beating heart surgery than on the design of the mechatronics system itself. These algorithms incorporate prediction, different sensing technologies, and advanced shared control paradigms required for a heart motion compensation system.

In [13], Bebek et al. introduce the Active Relative Motion Cancelling (ARMC) algorithm to cancel the relative motion between the surgical instruments and a point of interest on the beating heart, dynamically stabilizing the heart. This is a model-based algorithm which employs the novel use of biological signals, specifically the Electrocardiogram (ECG), to achieve motion cancellation. The motion tracking problem is rephrased as a reference signal estimation problem with the help of a model predictive controller. Taking advantage of the quasiperiodicity of the heart motion, the estimated signal is derived from the previous cardiac cycle. To compensate for errors introduced by heart rate variations, the ECG
wave forms are incorporated into the controller to adjust the tracking. It was noted by the authors that using the ECG signal would be insufficient for severe rhythm abnormalities, occurring due to arrhythmias for example. Bebek et al. also discuss the incorporation of other biological signals, including aortic, atrial and ventricular blood pressure.

These concepts are extended in [32], moving more towards prediction of the heart motion for their adaptive estimation algorithms. The proposed algorithm uses a specialized adaptive filter to generate future position estimates. A sonomicrometry system was used as the primary sensing technology. Based on a vector autoregressive model, recursive least squares is used to update the filter weights, with past observations being exponentially windowed. The predictive controller effectively implements Receding Horizon Model Predictive Control (RHMPC) in the feedforward path, and outperforms an EKF-based algorithm in terms of tracking performance. The EKF performed better in the presence of high-measurement noise and variations in the heart rate, however. The tradeoff in this method is between the noise in the estimator and the computational load.

A fast adaptive approach which can better handle extreme irregularities in the heart motion is presented by Liang et al. [33]. This adaptive nonlinear heart motion model is based on the Volterra Series. The Volterra Series is widely used in physiology signal estimation and modelling, naturally derived from the nonlinear dynamics of heart motion, offering good estimation of the phase-coupling phenomenon in heart motion dynamics. Liang et al. investigate these phase coupling effects between respiration and cardiac motion using Fourier and bi-spectral analyses. It was found that the nonlinear quadratic coupling regulates the dynamics of the beating heart and can improve the prediction accuracy by covering sharp change points resulting from even extreme irregularities in the heart motion.

Moving from predictive to force control, in Kesner et al. [34], 3D ultrasound guidance and position-modulated force control are used to command a robotic catheter system to apply a constant force against a moving target. The 3D US is capable of imaging and tracking both the catheter and the soft tissue, and these measurements are fed into an EKF that estimates the current tissue location based on a Fourier decomposition of the cardiac cycle. A clever force sensor is designed with a fiber optic transducer, and provides feedback to the controller which adjusts the catheter motion due to the forces experienced by the catheter tip while interacting with the moving target. Both Coulombic friction and backlash compensation models are employed in the controller, however the parameters of these models
are very system dependent.

The recent work described in [35] combines force feedback (does not require vision data), adaptive control and predictive control in a novel robotic control architecture for beating heart motion compensation. The controller consists of two cascade loops: the inner loop is model-reference adaptive control loop based on the Kalman Active Observer (AOB), and the outer loop is based on a Model Predictive Control (MPC) approach that generates control references. The AOB inner-loop imposes the desired and stable closed-loop dynamics, based on non-linear feedback linearization, augmented state-feedback, and stochastic design. The MPC outer-loop generates force references for AOB control by predicting the applied force in a finite time horizon. The robotic arm used in the validation study, a 4 DOF WAM arm from Barrett Technology, is equipped with a 6 DOF JR3 force sensor. This procedure neglects torsional ventricular motion.

While the work with force control presents promising results, the practical considerations of developing a surgically feasible force sensor that can be easily incorporated in \textit{in vivo} applications, survive sterilization procedures and still function in the messy clinical setting cast doubt on the clinical feasibility of these solutions. For the other control schemes, the challenge is to transfer these clever strategies to a plant/robot which will be stable with such a controller, and also allow the surgeon to perform complex tasks such as anastomoses.

### 2.3 Virtual Fixtures

Another interesting controls concept surrounding beating heart surgery is that of virtual fixtures, or active constraints. Virtual fixtures enable robots to intelligently collaborate with human users to complete tasks that are too complex for the robot to complete autonomously but too challenging for the user to complete unassisted. This is not a new concept, however the body of work on \textit{dynamic} virtual fixtures, specifically in the context of teleoperation and beating heart surgery is not extensive.

The first reported attempt to employ dynamic virtual fixtures for cardiac surgery was achieved by Park et al. [36]. In this work, the virtual fixtures were meant to assist in the artery harvesting portion of a robot-assisted CABG procedures. The virtual fixture consists of a forbidden region virtual wall/plane to help during a blunt dissection task of a stationary
target on a Zeus robot. *A priori* data was used to define the fixture.

Dynamic 3D virtual fixtures were proposed by Ren et al. [37] to augment their visual guidance system with haptic feedback. Pre-operative MR/CT images are used to generate the dynamic virtual fixtures, which are then mapped to the patient during surgery. This work combines two classes of virtual fixtures: guidance and forbidden-region virtual fixtures. The combination of these generates a potential field based fixture that renders anatomical constraints sufficiently rapidly to provide meaningful haptic feedback to the surgeon. The potential field is based on generalized Gaussian functions and generalized sigmoid functions, providing an easily adjusted protective area and continuous force feedback. The validation tasks were keeping the tool on the surface of a beating heart, as well as a simulated cutting/dissection task.

Gibo et al. [38] considered the implications of a moving virtual fixture, based on motion of the environment. Two methods of implementation are presented: predicted-position and current-position virtual fixtures. For their experiments, they assumed ideal motion tracking and prediction, and tried 1 DOF simple periodic, complicated periodic, and random motion paths. The experimental task was to keep the tool tip at a fixed distance beneath the surface of the phantom tissue.

In more recent work, Navkar et al. [39] demonstrate a guidance approach that uses real-time Magnetic Resonance Image (MRI) to assist the operator in manoeuvring an interventional tool safely inside the dynamic environment of a the cardiac left ventricle (for a TransApical - Aortic Valve Implant). This is the first work to show real-time guidance on the beating heart while not using pre-operative information. This is significant because the registration problem is nontrivial, especially when deformations in the soft tissue are significant. Using real time MRI, the images are processed and a safe access path is calculated, updating a virtual reality scene of the area of operation and driving a force feedback interface that is used by the operator for controlling the manipulator. Their experiments were performed off-line, but had real, pre-recorded MRI data and a virtual scanner. Their results show that acquiring the MR images is the bottleneck of this process, with the additional complications that interventional tools may distort image.

Another recent real-time approach is presented by Ryden et al. [40], though this approach uses RGBD data and streaming point clouds. Without *a priori* knowledge of the
heart geometry, an efficient method for enforcing forbidden region virtual fixtures is proposed by defining a spherical virtual fixtures (with a definable radius) around each point associated with the object of interest. The force on the master is calculated as a virtual coupling between the master position (Haptic Interface Point (HIP)) and the proxy. In their experiments, a Phantom device is used for the master, and an Xbox Kinect acquires the RGBD (red-green-blue-depth) data for a beating pig heart, generating a virtual fixture in real time. The experimental task was to stay a fixed distance from the surface of the heart. The feasibility of having a Kinect in clinical practice is doubtful, and the demonstrated radii for the spheres in the virtual fixture were in the order of 2 cm, which is an entire order of magnitude greater than the dimension of the coronary arteries involved in the anastomosis during CABG surgery.

The main issue in most of these implementations is that they do not demonstrate performing complex tasks on the heart surface, for example anastomosis for CABG. The over simplified tasks demonstrate the advantages of the proposed virtual fixtures, but in a clinically practical situation the surgeon will need to penetrate the surface of the heart to perform the anastomosis.
Chapter 3

Heart Motion

This chapter will provide a detailed explanation of the heart surface trajectory used for this work, as well as the technical difficulties of performing motion tracking of the highly dynamic and complex cardiac motion.

3.1 Heart Trajectory

The trajectory of the beating cardiac surface varies by location on the heart, the patient’s age, fitness level, and the severity of coronary disease. As the heart moves from diastole to systole in the cardiac cycle, the heart walls contract radially and longitudinally, and there is a wringing-like twisting motion of the left ventricle [41]. This corresponds to a complex, deformable motion model for a particular area on the surface of the heart with 3 degrees of translation and 3 degrees of rotation. For a given point on the surface, a 3 degree translational trajectory (i.e. rotation may be neglected) may be found, as in [42]. The trajectory of the heart surface is a function of the cardiac cycle, which is described as quasiperiodic, signifying that it is grossly periodic with slight changes in the amplitude or period length between periods. Thus, the motion of a point on the heart surface may be represented with a quasiperiodic waveform. It was assumed for this work that the heart motion is periodic. In reality, the heart motion is quasiperiodic, with the potential for arrhythmias. Clinically, beta-blockers can be used to regulate the primary heart frequency.

For the work described herein, the waveform reported in [42] was used. In [42], miniature radiopaque tantalum markers were implanted on the left ventricular midwall of donor hearts.
Figure 3.1: Direction of axes of heart motion [43].

at the time of cardiac transplantation. Stereo cineradiography allowed measurement of three
dimensional coordinates of multiple sites in the left ventricle roughly seven weeks after
surgery. The coordinate frame for the motion is defined as seen in Figure 3.1. Analyses were
performed for a point located on the left ventricular mid-wall, where the motion amplitude
is normally the highest. The maximum recorded displacements for this point in the x, y,
and z directions were 0.26cm, 0.59cm, and 0.97cm, respectively.

For the design of a robust system, worst case constraints must be considered. The trajec-
tory was therefore rotated such that the greatest variance was along one axis. This may also
be thought of as the trajectory which would be perceived from a well positioned robotic arm.
Principal Component Analysis (PCA) was used to obtain the rotation matrix which would
realign the eigenvectors of the trajectory with the principal axes defined in Figure 3.1. Since
the z-component was already the greatest in magnitude, the largest variance was aligned
with the z-axis. Further, the reported trajectory only consists of 35 recorded points for one
period. To eliminate any artificially high frequencies in the trajectory, a 10 Hz low pass filter
was applied to the motion. This is justified by spectral analyses reported in literature, which
demonstrate extremely small amplitudes above 10 Hz [11], [12]. Finally, for this work, the
heart motion was assumed to be periodic. A comparison of the translation components of
the original and rotated trajectories may be seen in Figure 3.2. A direct comparison of the
rotated trajectory components is seen in Figure 3.3. Finally, a three dimensional view of the rotated trajectory used in this thesis is seen in Figure 3.4. Note that this path is purely translational, i.e. for the superimposed control scheme only a translational offset is applied to the PSM while it is following the automated trajectory. The only rotation commands are issued from the MTMs.
3.2 Kinematics

To calculate the theoretical Cartesian velocities of the heart trajectory, the displacement for each time step was considered, and divided by a time step corresponding to a heart rate of 1.5 Hz, or 90 Beats per Minute (bpm). This number is based on heart motion measurements and analysis in [33] and [44]. This is again a slight overestimate, but designing for worst case scenario increases the safety margin of the system for a realistic setting. In clinical applications, specific chemicals (beta blockers) may be used to slow down and pace the heart during surgery. For

\[ x_i = [x_i, y_i, z_i]^T \]
\[ \dot{x}_i = [\dot{x}_i, \dot{y}_i, \dot{z}_i]^T \]
\[ \ddot{x}_i = [\ddot{x}_i, \ddot{y}_i, \ddot{z}_i]^T \]

the theoretical Cartesian velocities are calculated using

\[ \dot{x}_i = \frac{x_i - x_{i-1}}{dt} \] (3.1)
and the resultant theoretical maximum velocities are \( \dot{x} = [0.03, -0.12, 0.20] \text{ m/s} \). The instantaneous acceleration is calculated by

\[
\ddot{x}_i = \frac{x_i - x_{i-1}}{dt}
\]

(3.2)

for small dt, and the theoretical maximum instantaneous accelerations in each dimension are found to be \( \ddot{x} = [3.63, 10.91, 11.96] \text{ m/s}^2 \).

When implemented on the robot, the kinematics of the trajectory are calculated using the sensor feedback from each of the joints (more information about the robot is found in Chapter 4). The Cartesian position of the end-effector is calculated using forward kinematics and readings from encoders at each joint in the manipulator. Joint velocities, \( \dot{q} \), are also provided from the sensor hardware, and Cartesian velocities, \( \dot{x} \), are calculated using the inverse of the manipulator Jacobian, \( J \), following

\[
\dot{x} = J^{-1} \dot{q}
\]

(3.3)

at a specific, centered \( q \).

The acceleration is calculated using a second order digital Butterworth filter with a 10 Hz cutoff frequency to smooth any noise in the sensor readings. The transfer function coefficients were obtained using the Matlab function \texttt{butter}. A plot of the calculated Cartesian kinematics, using the joint encoder readings as input, for several periods of a heart trajectory may be seen in Figure 3.5. It may be observed that the maximum velocity and acceleration values seen in Figure 3.5 are significantly lower than the theoretical values calculated above. Recall that the raw heart trajectory data provided in [42] was filtered with a low pass filter with a cutoff frequency of 10 Hz before being commanded to the robot. The theoretical values are calculated using the raw data, but Figure 3.5 uses the filtered data. The reduction of the maximum acceleration verifies the filtering procedure.

### 3.3 Spectral Analysis

Finally, a spectral analysis of the heart motion was performed to evaluate its frequency components and harmonics, shown in Figure 3.6. A discussion of how these harmonics are incorporated in the control design may be found in Chapter 5.
Figure 3.5: Kinematics of heart trajectory.

Figure 3.6: Spectral analysis of heart trajectory.
Chapter 4

Platform Used / The System

One of the primary novelties in this thesis work is the implementation of the heart motion compensation system on an already clinically prevalent platform, the da Vinci® Surgical System by Intuitive Surgical Inc. This work was implemented in such a way that the master interface is not modified, thus the surgeons’ existing familiarity and experience with the robot may be leveraged. Further, the da Vinci has already received FDA approval and has become the standard of care for certain procedures. Thousands of systems may be found in hospitals worldwide. Its clinical acceptance for soft tissue applications makes it a perfect platform on which to target more advanced applications. While a complete heart motion compensation is very complex and would face significant challenges in receiving approval from both the FDA and the community of surgeons, implementing such a system on the da Vinci would mitigate some of these challenges, and increase the feasibility of achieving clinical practice.

Despite its advantages, there are still a number of difficulties involved with using such a complicated robotic platform. This chapter will provide a detailed description of the system, outlining these difficulties, but also discussing some of the benefits of using the da Vinci as a platform for this minimally invasive heart motion compensation system.

4.1 The da Vinci Research Kit

The dVRK, as discussed in Section 1.2, is an open source mechatronics platform connected to components of a da Vinci Classic (the first generation system produced by Intu-
itive Surgical). Developed by a collaborative effort between Intuitive Surgical Inc., Johns Hopkins University and Worcester Polytechnic Institute, the open source platform includes

- Mechanical components: robotic manipulators (da Vinci Classic), foot pedal tray, stereo viewer.

- Electronic hardware: controller boxes that include circuitry to communicate with the sensors and actuators in the robotic manipulators, as well as with the computer running the robot software [45].

- Software: cisst/SAW software libraries that provide access to all levels of control of the robot [46].

4.1.1 Mechanical Components

In the Robotics and Control Laboratory at the University of British Columbia, the dVRK setup includes a full da Vinci Classic system, as seen in Figure 4.1. This includes the master console (Figure 4.2) and the Patient Side Manipulators (PSMs) complete with setup joints (Figure 4.3). The master console consists of two MTMs, left and right for bimanual teleoperation, the stereo viewer for a 3D view of the surgical scene, and a foot pedal tray which
Figure 4.2: The Master Console of the da Vinci Classic

Figure 4.3: The Patient Side Manipulators of the da Vinci Classic
allows for digital user inputs to the system. The patient side includes three PSMs and one Endoscopic Camera Manipulator (ECM), as well as the full setup joints that allow for convenient positioning of each manipulator. On the PSMs, each joint is independently actuated with a Maxon DC motor, and joint position is measured using a rotary potentiometer and a complementary laser rotary Canon encoder.

4.1.2 Electrical Hardware

There is one controller box pertaining to each manipulator, totalling six active controllers in the stack at UBC, seen between the master console and patient side cart in Figure 4.1. Inside each box, shown in Figure 4.4, there are two custom designed IEEE-1394 Motor Controllers. Each motor controller consists of a Quad Linear Amplifier (QLA) board and Field Programmable Gate Array (FPGA) board. Together, the pair of motor controllers is capable of controlling eight motors, hence one box per manipulator. Each manipulator connects to its respective controller box via a zero insertion force DL156 pin connector and a break-out board provided by Intuitive Surgical. Controllers are connected in a daisy-chain structure using a IEEE-1394a firewire bus that weaves externally and internally to the boxes. A safety chain is also established, allowing for an emergency stop to immediately open the circuit and kill power to the manipulators.

The DL156 connector sends current commands from each controller box to the motors, and communicates the sensor readings back to the controllers. Through the IEEE-1394a
chain the sensor readings for each manipulator are sent to the Linux PC performing the high level control, where the next motor current commands are computed and sent to the control boxes via the same IEEE-1394a connection.

4.1.3 Software

The architecture of the system is designed for centralized computation and distributed I/O, meaning the majority of the control is implemented on a Linux PC and high speed communications to the I/O to each of the controller boxes distribute the controls to the mechanical components. More details about this architecture may be found in Section 5.3. A low level C++ Application Programmer’s Interface (API) is provided to perform data read/write through the I/O (writing desired motor currents and reading sensor data using the IEEE-1394a firewire protocol). The most recent revision of the open-source firmware employs the broadcast communication protocol in which asynchronous read/write operations require about 35μs, allowing control rates up to 6 kHz. Finally, a C++ component-based architecture with convenient inter-component interfaces allows for the implementation of the joint level servo loop control and high level robot control algorithms on the Linux PC.

4.2 Control Architecture

The control architecture of the dVRK was developed by Johns Hopkins University, and was used as the foundation for this work. The automated follow trajectory module was built directly on top of Hopkins’ infrastructure. The control architecture for teleoperation on the dVRK consists of three levels of control implemented in the software modules, and one level within the hardware. The lowest level of control resides within the FPGA verilog code which performs current control for each of the four motors controlled by a QLA/FPGA pair. Joint velocity estimation and motor current safety checks are also performed at this level. Having this control at the hardware level is the foundation for the centralized computation and distributed I/O architecture.

The firmware available on the QLA/FPGA controllers allows for broadcast communication protocols via the IEEE-1349a bus between the hardware and the Linux PC responsible for the remaining levels of control. The lowest level of control logic on the software side runs a joint level servo PID control loop. The software thread on which this control is com-
puted is tied to the I/O thread, minimizing the latency and complications that could arise from stale data caused by asynchronous threads. The I/O software modules communicate with the hardware to read data from the robot sensors and write data to the robot actuators directly through the IEEE-1394a connection. This information is shared with the software layer containing the joint level servo PID control loops; there is one joint level servo PID component per manipulator. In this layer, the desired joint positions and current joint positions are used to compute the joint level torques using a PID control scheme, for each manipulator. While the first three DOF on each PSM are independent, the last four are coupled nontrivially, so a joint-to-actuator mapping (instrument specific and provided by Intuitive Surgical) is necessary to convert desired joint torques to actuator torques.

The next layer of logic encapsulates the robot arm objects, with one component per MTM or PSM. These components are responsible for computing the forward and inverse kinematics for the manipulator. More details about this are in Section 4.3. At this level, the desired Cartesian position is translated into a desired joint position and sent to the joint level servo PID control layer. Further, this level also includes a gravity compensation mechanism on the MTMs, implemented using the Recursive Newton Euler Algorithm (RNEA). RNEA determines the accelerations and set of joint torques requires to compensate for manipulator dynamics, given the joint positions and velocities. Through system identification experiments performed by Omid Mohareri (a past PhD student working on the dVRK), all the MTM dynamic parameters (including link masses and inertia) and some of the Coriolis and centrifugal parameters for each arm for the da Vinci at UBC were roughly identified and incorporated into the model-based RNEA [47].

All these lower levels of control have been defined by the Johns Hopkins group. The highest level of control includes the teleoperation and automated trajectory following modules. The trajectory following is my own contribution, built on top of the provided framework. The basic teleoperation control structure was established by Hopkins. In the case of the motion compensation system, the teleoperation commands are superimposed upon the automated trajectory being executed by the PSMs. At this layer, the surgeon commands the masters to indicate the desired location for the PSM, and this master position is transformed to determine the commanded slave Cartesian positions. These desired positions are sent down to the robot arm objects through the component interfaces. The control loop frequencies for each of the software levels of control for the UBC system were determined either by Omid Mohareri or myself, and are described in Table 4.1. Note that the software
Table 4.1: Control loop frequencies

<table>
<thead>
<tr>
<th>Control Level</th>
<th>Loop Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O and joint-level PID</td>
<td>1 kHz</td>
</tr>
<tr>
<td>Forward/inverse kinematics</td>
<td>333 Hz</td>
</tr>
<tr>
<td>Automated trajectory generation</td>
<td>333 Hz</td>
</tr>
<tr>
<td>Teleoperation</td>
<td>200 Hz</td>
</tr>
</tbody>
</table>

is not implemented in hard real time, so the actual frequencies may vary.

4.3 Robot Kinematics

The MTMs are 8 DOF actuated manipulators while the PSMs are 7 DOF actuated manipulators. Both are equipped with joint sensors at each joint: a relative, high resolution laser rotary Canon encoder and an absolute, lower resolution rotary potentiometer. The primary focus of this thesis is to superimpose automated trajectories on the teleoperation commands for regular use, so while the performance of the MTMs will be the same as in clinical use, the behaviour of the PSMs will be semi-automated. Thus, the kinematic and dynamics properties of the PSMs are of greater importance.

The seven joints of the PSM are shown in Figure 4.5. The first three joints are the outer yaw of the arm, outer pitch of the arm, and insertion of the adapter, respectively. The
insertion joint is the only prismatic joint, the remaining six are all revolute. The last four joints correspond to the outer roll, wrist pitch, wrist yaw 1 and wrist yaw 2, all pertaining to the dexterous control of the instrument attached to the PSM. The design of the PSM is such that it is mechanically constrained to respect a Remote Center of Motion (RCM), as seen in Figure 4.6. This fulcrum point is invariant to the configuration of any of the actuated joints in the PSM, and may only be relocated by moving the setup joints.

To compute the forward and and inverse kinematics of each manipulator, Intuitive Surgical has shared a confidential User Guide for the da Vinci Research Kit with the dVRK community. The guide contains the kinematic parameters of the manipulators using the Denavit-Hartenberg (DH) convention. This convention describes how coordinate frames are attached to the kinematic chain of the robotic manipulator, moving from the base to the tip. It also defines the parameters to determine the transformation between coordinate frames, namely $a$ - the link length, $\alpha$ - the link twist, $d$ - the link offset, and $\theta$ - the joint angle. Using this information the forward and inverse kinematics of each manipulator may be calculated.

The coordinate frame of interest is that of the tip of the kinematic chain, located at the end of the end-effector of the attached instrument, O6 in Figure 4.7 just above the jaws of the instrument. The jaws for each instrument are different, so Intuitive selected a more consistent location for which to provide the Cartesian position. Also of note in Figure 4.7 is the orientation of the coordinate frame at O6. The x-axis corresponds with the outer yaw
Figure 4.7: DH Frames for the PSM and attached (Large Needle Driver) instrument [48]
of the PSM, the y-axis corresponds to the outer pitch, and the z-axis corresponds with the insertion joint. The axis with the greatest displacement during the heart trajectory is the z-axis, signifying the insertion joint will be under the greatest scrutiny for the analysis of the dVRK’s capability of tracking the heart motion. The PSM is a complex, cable-driven robotic mechanism with intricate dynamic effects. Specifically, the insertion joint, actuated by a cable-driven counter-weight mechanism, demonstrates highly nonlinear friction between its actuated joint limits. This introduces additional complications for controller design, discussed further in Chapter 5.
Chapter 5

Controller Design

This chapter will discuss the design and development process for the controllers employed to try to minimize tracking errors when commanding the dVRK PSMs to track the periodic motion of a beating heart. For more information about the kinematics and dynamics of the PSMs please refer to Chapter 4, and for further details about the heart motion please look to Chapter 3. As a very quick overview of the heart trajectory, the desired position signal of a point on the heart surface is a three dimensional, quasiperiodic motion rotated using PCA such that the maximum variance occurs along one axis, corresponding to the insertion joint on the PSM. The principal harmonic of the heart motion occurs at 1.5 Hz, and the trajectory experiences an amplitude change of nearly 12 mm in the z-direction. For motion tracking, the trajectory has been assumed to be period, and filtered with a low pass filter with cutoff frequency of 10 Hz.

5.1 System Identification

One of the primary contributions of this thesis is the use of the da Vinci (specifically the dVRK controlling a da Vinci Classic system) as the platform for a heart motion compensation system. The preliminary research question is whether the da Vinci Classic PSMs are even capable of tracking the high amplitude, high frequency heart trajectory, or whether its complex dynamics and nonlinear friction will prevent it from achieving the required accuracy. In order to develop a control system for heart motion compensation, the first step is to determine a model, or an analytical representation for the plant to be controlled, usually in the form of a transfer function. The transfer function is a representation of a linear time
invariant dynamical system, describing the relationship between the inputs and outputs of the system [49]. The process by which this transfer function is determined is called system identification.

There are two main types of approaches to performing system identification: first principles and empirical [50]. The first is a theoretical approach which employs fundamental laws of physics. The latter approach, also called data-driven modelling, is based on analyses of experimental observations about the system. The first principles approach requires a very thorough understanding of the physics present in the system, which is less practical for the nonlinear, dynamically complex 7 DOF da Vinci PSM. The data-driven approach is therefore used here to determine a model for the manipulator. Many techniques are available to perform the data-driven system identification, including analysis of the step response, frequency response, or impulse response, to name a few. Further considerations include incorporating the gearing in the drive train, friction modelling, and taking care not to hit any saturation limits – including physical joint limits, velocity or torque limits, etc. – as this will introduce nonlinearities to the system.

As an initial strategy, experiments were performed to try to find the transfer function describing the motors on the PSM, between the joint level torque, $\tau$, and joint position, $q$, as seen in the block diagram in Figure 5.1. The theoretical dynamic equation of a manipulator characterizes this relationship between the torque and manipulator motion. Through the Lagrangian equation [51], the following closed-form solution is found to be
Figure 5.2: Block diagram representation of the PSM motor plant controlled by PD ($K_I = 0$).

$$M(q)\ddot{q} + C(q, \dot{q}) + G(q) = \tau$$

(5.1)

where $q$ is the $7 \times 1$ vector with the joint angles for the 7 DOF PSM, $M(q)$ is the $7 \times 7$ inertia matrix, symmetric and positive definite for all values of $q$, $C(q, \dot{q})$ is the $7 \times 1$ centripetal and Coriolis force vector, $G(q)$ is the $7 \times 1$ gravity loading vector and $\tau$ is the $7 \times 1$ torque vector.

Various approaches were considered, including trying to train neural networks to determine the inverse dynamics of the manipulator, or trying to incorporate fundamental equations such as Newton’s equations of angular motion or basic DC motor governing laws. The complexities and nonlinearities in the system made it extremely challenging to obtain a good model. Particularly for the insertion joint, the friction introduced nonlinearity and unrepeatable hysteresis. Commanding a sinusoidal torque to the motor, the drift in the joint position was clearly evident. The experienced friction was dependent upon on the velocity of the joint, the frequency of the sinusoid, and the joint position, among other factors.

The reminder this section describes the process actually used, and an edited description of this process may be found in [52].

The plant was therefore redefined to contain the motor plant $P$ as well as a tuned joint level Proportional-Derivative (PD) controller, $C$, with unity feedback, as seen in Figure 5.2.
The closed loop transfer function describing the plant is now

\[ H(s) = \frac{CP}{1+CP} \]

After tuning the PD following the Ziegler-Nichols method ([53]), tracking errors in the order of 1-2 mm were achieved between desired and measured Cartesian position. More importantly, the controlled system was now linear, and when sinusoidal inputs were provided to the plant sinusoidal outputs would be observed. To continue with the empirical system identification process, experiments were conducted to gather data about the inputs and outputs of the plant at various frequencies. In future work a nonparametric model could be found using Matlab’s `tfestimate` command.

Since the heart motion compensation system is designed to compensate for only the three dimensional translational components of the heart trajectory, system identification only needed to be performed on the first three joints of the PSM. The first three joints are outer yaw (x), outer pitch (y), and insertion (z) and the joints are independent of each other. Note the Cartesian motions are coupled since the instrument has a finite length; in other words, if a pure outer yaw motion is commanded the end effector of the instrument will travel in an arc with varying z-position since the PSM has a fulcrum point at the RCM.

For each of the first three joints, the closed-loop frequency response between the commanded and measured Cartesian positions of the tool tip was found. The generalized form for the parametric model is a second order transfer function

\[ H(s) = \frac{K(1 + Ts)}{1 + 2\zeta T_\omega s + (T_\omega s)^2} \]  

(5.2)

In Equation 5.2, the zero corresponds to the differential term in the PD controller and the two poles correspond to the inertia and damping which govern the motor dynamics of the manipulator. To experimentally determine these parameters, the system was excited by a position signal composed of a summation of sinusoids. Using *a priori* knowledge about the frequency spectrum of the heart trajectory, the performance of the system was investigated at frequencies corresponding to the spectral frequencies of the heart trajectory. To be consistent with the approach for the spectral line controller (discussed in Section 5.4), the harmonics with the four greatest amplitudes were used, such that the excitation signal
Table 5.1: Experimentally Determined Transfer Function Parameters for PSM Joints

<table>
<thead>
<tr>
<th></th>
<th>$K$</th>
<th>$T_z$</th>
<th>$T_\omega$</th>
<th>$\zeta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outer Yaw</td>
<td>0.8852</td>
<td>-0.0118</td>
<td>0.0600</td>
<td>0.3291</td>
</tr>
<tr>
<td>Outer Pitch</td>
<td>0.8917</td>
<td>-0.0150</td>
<td>0.0573</td>
<td>0.2906</td>
</tr>
<tr>
<td>Insertion</td>
<td>1.0006</td>
<td>-0.0069</td>
<td>0.0100</td>
<td>1.3155</td>
</tr>
</tbody>
</table>

for each joint was

$$x = \sin(f_1t) + \sin(f_2t) + \sin(f_3t) + \sin(f_4t)$$

$$f = [1.5, 3.0, 4.5, 7.0] \text{Hz}$$

The commanded positions for the other two joints remained fixed at zero. The Matlab System Identification Toolbox was employed to find the parameters of the model using the provided input-output data. The parameters are summarized in Table 5.1. Each PSM was thus characterized as a linear combination of three independent Single Input Single Output (SISO) plants.

The simplifying assumption of a second order system will introduce some error, particularly when the plant is inverted. This assumption is considered acceptable for this feasibility study since submillimetre tracking errors were still achieved (see Figure 5.10 and Figure 5.11). In future work, nonparametric data-driven system identification methods will be employed to eliminate this simplifying assumption.

5.2 PID Control

This section will provide a brief description of one of the most common control schemes, PID control. A block diagram representation of a PID controller is shown in Figure 5.3. The controller accepts as input the tracking error, $e(t)$ which is the difference between the desired set point, $r(t)$ and the current measured process variable/output of the plant, $y(t)$. A series of calculations are performed using the tracking error to compute the value of the control signal, $u(t)$ to be input to the plant. The control algorithm depends on three independent parameters: the proportional gain, integral gain, and derivative gain, denoted
Figure 5.3: Block diagram of a PID controller

$K_P$, $K_I$, and $K_D$, respectively, in Figure 5.3. Conceptually, the proportional gain tries to minimize the present error, the integral gain considers the accumulation of error, and the derivative tries to prevent future errors. The weighted sum of these three computations determines the control input, $u(t)$ for the plant. This controller is widely popular because it relies purely on the measured output of the plant, not on an understanding of the underlying physics and properties of the plant being controlled [54]. Moreover, the common assumption is that the plant behaves more or less like a second order (spring-mass-damper) system in the frequency range in question, which simplifies the modelled plant dynamics.

A well-tuned PID controller can be largely successful for a large variety of systems, so the process of tuning the controller remains the greatest challenge of PID control. There are a number of established methods for tuning a controller, such as manual tuning or the Ziegler-Nichols method [53]. Typically more aggressive gains will minimize tracking error and rise time/lag, however the trade off with overshoot, settling time, and most importantly stability must be considered carefully. These considerations vary depending on the application.

For this work, there is joint level PID control, meaning the PID parameters for each joint were optimized. Even so, the heart trajectory is highly dynamic, and attempted experiments concluded that tracking errors less than 1.5 mm could not be achieved with a simple PID controlling the dVRK PSM. The vessels involved in the target application of anastomosis for CABG surgery are around 2 mm in diameter, so submillimetre tracking error is required for clinical feasibility.
5.3 Simulink to C++ Interface

While making efforts to design and develop more advanced controllers for heart motion tracking, it was observed that the controller design and development process in the cisst/SAW framework of the dVRK was quite tedious. The cisst/SAW software architecture for the dVRK is wonderfully engineered, however it is a huge collection of code written in C++ requiring several minutes to compile. Controller design was being done in the Matlab Simulink environment, utilizing SISOTool and various other useful toolboxes for control system design and analysis. The design would then have to be discretized and implemented in C++, integrated into the cisst/SAW software stack, and then usually debugged. Each iteration of design was very time consuming. In an attempt to expedite the process, an interface between Simulink to the C++ cisst/SAW was developed. With this interface, the process variables would be communicated from the cisst/SAW code to Simulink, the designed control algorithm would compute the process input, specifically joint torque, and then communicate this value back to cisst/SAW to command the robot. The user could control whether the default joint level PID control in the cisst/SAW code or the Simulink control has exclusive command of the robot. Effectively outsourcing the control logic to Simulink, this would allow the control designer to change the controller and test it on the robot without recompiling or power cycling the robot.

This work was submitted as an abstract ([55]) and presented as a poster entry at the International Conference for Robotics and Automation (ICRA) 2015 in Seattle, Washington, at the workshop discussing Shared Frameworks for Medical Robotics Research [1], where it was the runner up for best poster. The abstract follows in this section.

5.3.1 Introduction

The da Vinci Research Kit (dVRK) is an open source mechatronics platform – consisting of hardware, firmware, and software – that serves as a common platform for research in medical robotics, specifically in the area of telesurgery [46]. The platform has been specifically designed to grant researchers access to all levels of control of the robot. A centralized computation and distributed I/O [46] architecture is used to achieve low-latency interfaces between the hardware and software, reduce cabling, and allow developers to implement their software in a familiar development environment on a high-performance Linux PC. The PC is connected to I/O hardware (sensor and actuator readings) via an IEEE-1394
There are currently approximately 17 institutions included in the user group in the dVRK research community [59]. These groups study many facets of teleoperation, including haptic feedback, semi-autonomous/autonomous control, and gaze tracking. In order to implement a controller required for such projects, the typical procedure for controller development consists of three major stages: (i) controller design (ii) implementation and (iii) testing and debugging. A highly popular tool for controller design is MATLAB® Simulink®, a block diagram environment containing many useful toolboxes and flexibility for modelling, simulation, and controller evaluation [60]. Simulink also offers the ability to “peek and poke” at signal values in the model, a highly useful tool in the controller development process. To implement a controller in the cisst/SAW dVRK platform, a completed controller design must first be discretized then integrated into the C++ cisst/SAW software. For debugging, user interface components and logging are available for use in the cisst/SAW framework; however, applications for analyzing this data must be written by the researcher.

In this work, we propose a novel interface between Simulink and C++ which will allow the control logic to remain in the Simulink environment for design, easy block diagram implementation, evaluation, and debugging. With the ability to switch between the native joint-level PID controller in the cisst/SAW software and the custom Simulink controller, the researcher now has the ability to try new controllers “on the fly”. This will serve as a tool for rapid-prototyping controller designs without having to power off any hardware, write any C++ code, or compile the large cisst/SAW stack for each modification.

5.3.2 Implementation

5.3.2.1 Preserving the Component-Based Architecture

A component-based control architecture is used in the cisst/SAW libraries, with well defined levels of control and required/provided interfaces between components (shown in black in Figure 5.4). With the aim of making the Simulink to C++ interface abstract enough to be able to connect it at any level of control in the software, a new component,
Figure 5.4: Modified control architecture for one arm using TCP/IP based Simulink to C++ interface. The logic to switch between controllers is described in 5.3.2.4. Blue indicates the new components and interface, black indicates the original control architecture in the cisst/SAW framework from [46].

*mtsSimulinkController*, was created. Other components may interface with the mtsSimulinkController using the standard required/provided interfaces available in cisst/SAW. As a proof of concept test, the first Simulink dVRK controller that was implemented mimicked the default joint level PID control. In this example, the mtsSimulinkController block can directly replace the mtsPID block, as shown in Figure 5.4.

When the mtsSimulinkController is enabled, the mtsPSM component sends the desired position information to the mtsSimulinkController via the standard required/provided cisst/SAW interface. The mtsSimulinkController reads in the current position and velocity information from the robot IO, and then packages all the kinematic information into a custom data packet. This packet is then serialized and sent over a TCP/IP socket to the Simulink model. Handshaking is incorporated, and mtsSimulinkController will not send another packet until a response has been received. The received packet is accepted over a second TCP/IP socket and contains the joint level torque values which are commanded directly to the mtsRobotIO1394 component. As an analogy to the ROS communication protocol, mtsSimulinkController is a publisher of the kinematic data and subscriber of the torque data.
5.3.2.2 Threading

As an implementation detail, it is noteworthy that for smooth control of the arm, the mtsSimulinkController and the IO component must be tied to the same software thread. This is demonstrated between the mtsPID component and the IO component in the open-source cisst/SAW libraries. Two components may be assigned to the same thread using the ExecIn/ExecOut interfaces provided in cisst/SAW component code. This is necessary for the controller components since it guarantees a deterministic order for reading data from the robot’s sensors, computing a torque, and writing that torque to the robot actuators. If implemented on different threads, data synchronization issues arise, latent torque values are commanded, and a significant jitter may be observed in the robot’s trajectory.

5.3.2.3 Communicating with Matlab

Simulink 2012b was used for the development of this work. This version of Simulink does offer rudimentary support for TCP/IP communication in the form basic send/receive blocks. However these are not nearly flexible enough for this application, where custom data packets and serialization are required. As such, specially-designed s-function blocks were generated (Figure 5.5 and Figure 5.6). The s-function block encapsulates a C++ mex-function: a special C++ s-function file was written for each of the send and receive tasks. Each task has unique buffer size and serialization/deserialization requirements. To simplify data synchronization and timing issues, two TCP/IP sockets are employed, each unidirectionally, one strictly for the kinematic data (from C++ to Simulink), and one strictly for the torque data (from Simulink to C++). In the initialization of the s-functions (performed when the Simulink simulation is commenced), the two client sockets (the kinematicDataSocket and the torqueSocket) will connect to two server sockets created in mtsSimulinkController. The initiation of this process is controlled by the user through a check box in the graphical user interface. The control will not be passed to the Simulink unless both TCP/IP connections are successful. For every iteration in the simulation,

- The msfcn_acceptKinematicDataC block will read from the kinematicDataSocket, deserialize the data and parse it into vector outputs for the Simulink model to use (<NumberOfJoints,1> vectors for currentPosition, desiredPosition, and currentVelocity)
- The controller model uses Simulink blocks to take this kinematic data and compute
Figure 5.5: Custom Simulink block for receiving data in Simulink/C++ interface. The length of the output vectors can be customized.

Figure 5.6: Custom Simulink block for sending data in Simulink/C++ interface. 

\[ \text{torque is a } <\text{NumberOfJoints, 1}> \text{ vector.} \]

- The required torque value (\(<\text{NumberOfJoints, 1}>\text{ vector})

- The torque values are serialized and written to the \text{torqueSocket} through the msfcn\_sendTorqueDataC block.

5.3.2.4 Switching Control

With the goal of rapid-prototyping, it is necessary to be able to switch to/from the Simulink on the fly. To guarantee safety, the default joint level PID controller is turned on whenever communication with Simulink is terminated (either due to a communication error, the simulation ending, or user-defined termination through the graphical user interface). As soon as communication with Simulink is established, the cisst/SAW joint-level PID is disconnected.
5.3.3 Analysis

As an initial proof of concept, two different controllers were developed. First, the default joint-level PID controller available in cisst/SAW was recreated in a Simulink model (Figure 5.7). Second, a Cartesian controller was developed. To test both, a single slave arm was automated to follow a trajectory. For the dVRK system at the Robotics and Controls Laboratory at the University of British Columbia (described in detail in [47]), we run the control loop frequencies at 1 kHz for the joint level PID and I/O level control; 333 Hz for the kinematics and robot logic control, and 200 Hz for the teleoperation control loop for bimanual teleoperation (controlling four robotic arms). Performance of the Simulink interface will be considered acceptable if the same frequencies can be maintained while the Simulink controller is commanding the trajectory on the slave arm.

Both the cisst/SAW code and MATLAB Simulink were running on the same Linux PC. Table 5.2 summarizes the timing analysis for the experiment from the perspective of the cisst/SAW code, where \( \mu \), \( \tilde{t} \), and \( \sigma \) represent the mean, median, and standard deviation of the specific measurement, respectively. Two measurements for each value are recorded, one for all data, one with the top 5% of the population removed, to eliminate outliers. This table is written from the perspective of the cisst/SAW software, so kinematic data is sent, and torque data is received. The loop time describes a simulation iteration in Simulink; it is measured as the elapsed time from the moment before the kinematic data is sent to the moment right after the corresponding torque data is received. This measure includes the read/write operation times over the TCP/IP sockets, serialization/deserialization of the packets (in both C++ and Simulink), computation of the torque values in Simulink, as well as the time spent waiting for a handshake confirmation. The sample time for the thread – as controlled by a lower level, timer-based component manager in the cisst/SAW code – is listed at the bottom of the table. These results demonstrate that the communication and processing offloaded to Simulink take only a small portion of the 1 msec period allotted for the IO and controls. The 1 kHz control loop frequency for both the joint level PID and Cartesian controller was successfully achieved when interfacing with the Simulink controller.

Occasionally during the experiments it was observed that Simulink would irregularly and temporarily experience a brief delay. Besides the robustification provided by the handshaking, additional safety features have been included to automatically switch back to the default mtsPID controller if the communication interruption is too long. The exact cause for these
Figure 5.7: Joint-level PID for a slave arm on the dVRK implemented as a Simulink model. The *TCP/IP Receive Data* subsystem on the left contains the block in Figure 5.5 and reroutes the signals to 7 \(<3,1>\) vectors, one for each joint. The *TCP/IP Send Torque* subsystem on the right consolidates the 7 torque values into a single vector and feeds this into the block in Figure 5.6.
Table 5.2: Timing analysis for Simulink joint-level PID (PID_θ) and Cartesian (PID_χ) controllers

<table>
<thead>
<tr>
<th>Measurement</th>
<th>PID_θ</th>
<th>PID_θ (95%)</th>
<th>PID_χ</th>
<th>PID_χ (95%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bytes per sent packet</strong></td>
<td>326</td>
<td>-</td>
<td>452</td>
<td>-</td>
</tr>
<tr>
<td>Sending data μ_t [msec]</td>
<td>0.0379</td>
<td>0.0306</td>
<td>0.0379</td>
<td>0.0318</td>
</tr>
<tr>
<td>Sending data τ_t [msec]</td>
<td>0.0283</td>
<td>0.0282</td>
<td>0.0317</td>
<td>0.0315</td>
</tr>
<tr>
<td>Sending data σ_t [msec]</td>
<td>0.0391</td>
<td>0.0081</td>
<td>0.0357</td>
<td>0.0053</td>
</tr>
<tr>
<td><strong>Bytes per received packet</strong></td>
<td>69</td>
<td>-</td>
<td>54</td>
<td>-</td>
</tr>
<tr>
<td>Reading data μ_t [msec]</td>
<td>0.0022</td>
<td>0.0016</td>
<td>0.0013</td>
<td>0.0008</td>
</tr>
<tr>
<td>Reading data τ_t [msec]</td>
<td>0.0011</td>
<td>0.0011</td>
<td>0.0008</td>
<td>0.0007</td>
</tr>
<tr>
<td>Reading data σ_t [msec]</td>
<td>0.0027</td>
<td>0.0020</td>
<td>0.0025</td>
<td>0.0002</td>
</tr>
<tr>
<td><strong>Loop time μ_t [msec]</strong></td>
<td>0.1514</td>
<td>0.1401</td>
<td>0.1867</td>
<td>0.1768</td>
</tr>
<tr>
<td>Loop time τ_t [msec]</td>
<td>0.1395</td>
<td>0.1391</td>
<td>0.1845</td>
<td>0.1817</td>
</tr>
<tr>
<td>Loop time σ_t [msec]</td>
<td>0.0816</td>
<td>0.0274</td>
<td>0.0698</td>
<td>0.0254</td>
</tr>
<tr>
<td><strong>Sample time μ_t [msec]</strong></td>
<td>1.0631</td>
<td>1.0516</td>
<td>1.0626</td>
<td>1.0526</td>
</tr>
<tr>
<td>Sample time τ_t [msec]</td>
<td>1.0559</td>
<td>1.0552</td>
<td>1.0561</td>
<td>1.0553</td>
</tr>
<tr>
<td>Sample time σ_t [msec]</td>
<td>0.1205</td>
<td>0.0402</td>
<td>0.1138</td>
<td>0.0335</td>
</tr>
</tbody>
</table>

delays is still under investigation, but it is suspected that having a PC dedicated to running the Simulink controller, instead of having one PC running both Simulink and the cisst/SAW application, would help alleviate this issue. Another potential cause of the delays is that the Linux PC used for the experiment was a non-hard real-time operating system.

A potential alternative solution for the interface would be to integrate UDP instead of TCP/IP sockets. Many resources discuss the comparison between these protocols. Ultimately, TCP/IP was selected for its reliability. While the extra handshaking between mtsSimulinkController and Simulink adds additional robustness, the data integrity guaranteed by TCP/IP transmission was deemed beneficial for this surgical robotics application. Also, UDP is a connectionless protocol; communication is datagram oriented. This would
make it more challenging to do safety monitoring of the integrity of the C++/Simulink interface. Instead, using TCP/IP, the connection status of the communication is immediately known, resulting in faster responses to error conditions. The most significant disadvantage with TCP/IP is that it is generally slower than UDP, but our experiment demonstrates that its performance is sufficiently fast for this application.

5.3.4 Conclusion

The Simulink to C++ interface described in this paper is a handy tool for controller development on the dVRK. This interface allows researchers to design, implement, and debug controllers all in the friendly MATLAB Simulink block diagram environment. Custom-written s-function blocks perform the tailored TCP/IP read/write functions. The Simulink controllers can be easily modified on the fly, and then be used to command the robot without having to recompile or relaunch the cisst/SAW code. The user specifies through a user interface when to switch control to/from the Simulink controller, and a safety switching mechanism will automatically switch to the default joint-level PID controller whenever the Simulink controller terminates. Further, the ability to “peek and poke” at signal values in the Simulink environment and monitor, store, and evaluate these signal values on the fly is a very helpful debugging tool. These advantages aim to simplify and expedite controller design for the dVRK. Once the controller design is completed, Simulink offers a C++ code generation tool which will transform the Simulink block model into C++ code which can then be easily integrated into the native cisst/SAW code.

Due to its modular nature, the general TCP/IP interface to/from Simulink may also be easily modified to communicate with other robotic platforms. This work will be packaged and prepared for distribution; its desired intent is to provide a controller development tool for the robotics community. The package will include the joint level PID and Cartesian controller examples. Please contact the corresponding author for details about the distribution.

5.4 Spectral Line Controller

Both this and the following sections have been previously described, in less detail, in [52].
As described in Section 5.2, the PID controller is capable of achieving good tracking results for following the heart motion, however better performance is required to achieve submillimetre tracking errors. For a more advanced controller, the quasiperiodicity of the heart trajectory is leveraged. (The spectrum and heart trajectory may be found in Chapter 3.) This work investigates the development of a spectral line controller to track the heart motion. The fundamental idea of the spectral line controller is to use the inverse of the transfer function at the heart motion spectral frequencies to compute a reference input that minimizes the tracking error. The published heart trajectory is used as prior knowledge to optimize the motion of the da Vinci PSM.

While spectral decomposition has been used before for periodic trajectory control [13], and proposed for heart motion tracking [61], it has not been used before for controlling a robot to follow heart trajectories. To accurately follow the heart trajectory, the Fourier spectrum of the target trajectory – the heart motion with a fundamental frequency of 1.5 Hz (consistent with [11] and [13]) – must be analyzed. The first step is to find the critical spectral lines and rank them by their amplitude. The frequency spectrum of the heart trajectory is shown in Figure 3.6 for reference, and Table 5.3 shows the extracted ranked harmonics.

The design tradeoff for the spectral line controller is between the complexity of the controller and performance. A greater complexity is introduced with a greater number of spectral lines. To determine how many spectral lines are needed for an adequate representation of the heart motion, the motion is reconstructed using the inverse Fourier transform of the heart trajectory was taken using the top $N = [1, 10]$ spectral lines. Figure 5.8 shows the total error over one period of the heart motion between the reconstructed trajectory and the original trajectory, depending on the number of spectral lines used for the reconstruction. It was

<table>
<thead>
<tr>
<th>Harmonic</th>
<th>Frequency</th>
<th>Amplitude [x,y,z]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.5 Hz</td>
<td>[52.18, 37.43, 1798.40]</td>
</tr>
<tr>
<td>2</td>
<td>3.0 Hz</td>
<td>[32.83, 212.20, 216.20]</td>
</tr>
<tr>
<td>3</td>
<td>4.5 Hz</td>
<td>[63.06, 178.31, 172.63]</td>
</tr>
<tr>
<td>5</td>
<td>7.5 Hz</td>
<td>[12.28, 84.79, 13.99]</td>
</tr>
<tr>
<td>4</td>
<td>6.0 Hz</td>
<td>[21.29, 67.23, 81.12]</td>
</tr>
<tr>
<td>6</td>
<td>9.0 Hz</td>
<td>[13.63, 22.72, 12.28]</td>
</tr>
<tr>
<td>7</td>
<td>10.5 Hz</td>
<td>[2.41, 4.48, 4.16]</td>
</tr>
</tbody>
</table>
Figure 5.8: Evaluation of heart motion reconstruction (total error over one period) versus number of spectral lines

Figure 5.9: Difference between original and reconstructed heart trajectories using $N = 4$ spectral lines

decided that using four spectral lines would provide a reasonable performance-complexity tradeoff, so the frequencies of $[1.5, 3.0, 4.5, 7.5]$ Hz were used further in this study. The difference between the original and reconstructed trajectory using $N = 4$ spectral lines can be seen in Figure 5.9.
To determine the pre-conditioned input trajectory that will minimize the tracking error with the spectral line controller, the **inversion principle** for a linear feedforward controller design is employed. The analytical formulation for the input is as follows:

\[ u_d(t) = \Re \left\{ \sum_{k=1}^{N} H(j\omega_k)^{-1} G(j\omega_k) e^{j\omega_k t} \right\} \]  

(5.3)

where \( G(j\omega_k) \) is the Fourier series of the heart motion at the \( k = 1, \ldots, N \) spectral frequencies, and \( H(j\omega_k) \) is the closed loop frequency response obtained in the system identification process described in [Section 5.1]. At each of the \( N \) harmonic frequencies, \( \omega_k \), the closed-loop transfer function is inverted and multiplied by the corresponding harmonic of the heart signal. The results are summed over the \( N \) frequencies, then the inverse Fourier transform is used to obtain the desired trajectory input \( u_d \) in Cartesian space along one dimension. This must be repeated for each dimension of each PSM used in the heart motion compensation system.

This pre-compensator worked well on the da Vinci PSM; the tracking results are summarized in Table 5.4 and Figure 5.10 and Figure 5.11. The first 0.2 seconds of initialization were not considered for the analysis. These results demonstrate that a heart motion tracking system on the da Vinci is feasible, verifying that the system is mechanically capable of achieving the dynamics required. Further, assuming the heart trajectory is periodic, tracking in open loop using this method can be sufficient if adjustments are made to the spectral line coefficients using a Kalman filter, as suggested in [61] for heart motion tracking, or other approaches.

From these results it was observed that a frequency component at 6.0 Hz was clearly evident in the tracking error signals, so incorporating this frequency into the computation of the pre-conditioned input would likely further reduce the tracking error. The greatest error is present in the y dimension, which corresponds to the outer pitch joint. During the

<table>
<thead>
<tr>
<th></th>
<th>X [mm]</th>
<th>Y [mm]</th>
<th>Z [mm]</th>
<th>Total Euclidean [mm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. Error</td>
<td>0.1882</td>
<td>0.5863</td>
<td>0.3859</td>
<td>0.7267</td>
</tr>
<tr>
<td>RMS Error</td>
<td>0.1147</td>
<td>0.3154</td>
<td>0.1090</td>
<td>0.3529</td>
</tr>
</tbody>
</table>

Table 5.4: Tracking Errors using Spectral Line Controller
system identification stage, when performing the data-driven modelling to fit the parametric second order transfer function, a measure of the quality of fit was provided. The measure for the outer pitch was less than for the outer yaw and insertion joints, which matches the tracking error results. Due to the sensitivity of the inversion principle, if the model of the plant is poor, i.e. the poles and zeros of the plant are inaccurately positioned, the resultant tracking performance may be made worse. Instead of cancelling out the effect of the plant dynamics through inversion, additional dynamics may be introduced which may be even more challenging to control. Moving towards a non-parametric model of the joints (using
data driven modelling technique and Matlab’s non-parametric system identification tools, for example) would eliminate some assumptions about the plant and might reduce some of the error in the y dimension.

5.5 Visual Stabilization

This work proposes a novel approach to virtual stabilization of the heart by decoupling the motion measurement and motion tracking tasks that are required in an actual system. A perfect motion measurement environment is simulated by mounting the camera in fixed relationship to the phantom that simulates the heart motion.

To provide a stable image to the console in a virtually stabilized environment, the camera must track the heart motion accurately either physically, through actual camera motion, or virtually, with 3D tracking and stabilized rendering. For preliminary studies, we propose here to decouple the visual tracking problem from the manipulation tracking problem by mounting a camera at a fixed distance to the payload that simulates the beating heart. Thus, for the study herein a small, light stereo camera was mounted to the robotic arm carrying the simulated heart target using a 3D printed fixture (seen in Figure 6.1 or Figure 6.2). Initially a prototype for the single-port da Vinci system, this camera (shown in Figure 5.12) is lightweight (about 8 g) with a small form-factor and provides standard definition stereoscopic views of the surgical site. This camera was interfaced to the da Vinci Classic master console so the surgeon is able to watch the camera feed in the actual stereo viewer in a realistic setting.

Figure 5.13 shows how the camera was mounted to achieve visual stabilization. Initially, for the proof of concept system, the camera was mounted with 3D printed plastic compo-

![Figure 5.12: Camera and 3D printed housing](image-url)
components directly to the instrument shaft, at a fixed distance from the heart target which was also mounted to the same shaft. In the second generation system, which included motion measurement, seen on the right, the camera is mounted to the tip of an instrument with a 3D printed plastic component again.

Figure 5.13: Camera mounts for visual stabilization
Chapter 6

Initial User Studies

With the system identification and spectral line controller design complete, and with the follow trajectory software module implemented, the system was ready to be evaluated by surgeons. User studies were performed to verify the initial, proof-of-concept system, with very promising results. A description of the system and user studies was submitted and accepted as a conference paper to ICRA 2015 [52], and has been described in further detail in the previous chapters. This system was also entered in the first Surgical Robot Challenge which took place at the 2015 Hamlyn Symposium for Medical Robotics, and the entry video may be found at [62]. This chapter will discuss the user studies performed to evaluate the proof of concept system.

As a quick overview, the system approach being evaluated consists of (i) using the auxiliary PSM to move a payload that simulates the heart surface motion, (ii) implementing tracking controllers to have the left and right PSMs track the payload, and (iii) demonstrating through standardized skill assessment tasks that teleoperation relative to the payload can be carried out without compromising task time and quality.

6.1 User Studies

The goal of the user study was to determine the change in task quality and completion time, when carried out by our motion compensating system on a payload simulating the beating heart, relative to the gold standard, which is the same task carried out on a stationary payload.
6.1.1 Setup

The participants sat at the da Vinci Classic master console and could see the stereoscopic camera feed in the native 3D viewer. They could interface naturally with the MTMs and clutch foot pedal to modify their workspace. Their teleoperation commands were superimposed upon the automated trajectories. At the slave manipulators, one PSM carried a target which simulated the heart surface while the other (right) arm performed the task. The heart surface targets were 3D printed fixtures mounted directly onto the shaft of the da Vinci large needle driver. A similar clamping mechanism was used to secure the camera to the same tool shaft above the platform. The pitch of the camera could be adjusted using a simple pin mechanism. The view of the camera could thus be adjusted with three degrees of freedom: height, roll around the tool shaft, and pitch angle. Two LEDs were mounted directly beneath the camera to help illuminate the scene (see Figure 6.1 - upper white part holds camera and LEDs).

Two different benchmark tasks were completed, each under the following five conditions completed in the same order:

- C1: everything stationary
- C2: moving target, no compensation
- C3: moving target, compensation enabled
- C4: moving target, compensation enabled with smaller amplitude motion
- C5: moving target, compensation enabled with slower frequency of motion

Condition 1 was designed as the control, analogous to robotic surgery on the stopped heart. Condition 2 simulates having to perform surgery on the beating heart, without any motion compensation assisting in the task. Condition 3 is the desired use case for full frequency, full amplitude compensation. Conditions 4 and 5 simulate potential solutions which can be employed should condition 3 be too difficult: smaller amplitude motion may be achieved using mechanical stabilizers, whereas condition 5 may be achieved using beta-blockers and heart pacing. Both solutions are clinically acceptable. For all the conditions, the camera was mounted at a fixed distance from the heart target, offering a visually stabilized view when the target was moving.
6.1.2 Participants and experiment completion

Five right-handed participants were included in the study: three cardiovascular surgical residents with basic da Vinci training, and two surgeons who are expert da Vinci users. One of the expert users is the only cardiac surgeon in central and western Canada to perform robotically assisted cardiac surgeries.

The users were given time to practice, and then three trials of each condition were executed. For conditions 2 through 5, the da Vinci slave arm(s) was(were) automated to follow a realistic pre-programmed heart trajectory. The virtually stabilized field was observed in conditions 3 through 5. The surgeon’s commands through the MTM were superimposed upon the automated trajectory of the right-hand arm. For conditions 1 and 2, the right slave arm performed the surgeon’s commands exactly, whereas for conditions 3 to 5 the surgeon’s motions were augmented with the heart trajectory. After completing all conditions for one task, the participants filled out a questionnaire evaluating their subjective perception of the difficulty of each condition for each task.

6.1.3 Task Design

The two benchmark, one-handed tasks that were selected for this study include a simulated suture and a pegboard transfer task. The simulated suture task fixture is shown in Figure 6.1. Following [10], paper targets with dot patterns were affixed to a 3D printed plastic frame such that the surgeon was able to pierce the paper with a needle at the printed dot locations. Working towards the ultimate goal of anastomosis, the first step is to verify that the surgeon can place the suture at the desired locations with sufficient precision. The dots had a diameter of 1 mm and the needle’s diameter was 0.65 mm. A target was “hit” if the error was \(<0.5\) mm, and “missed” if the error was \(\geq1.0\) mm. Successful completion of this task requires high accuracy, similar to what would be expected for actual anastomosis of a coronary artery. The users were instructed to pierce the dots one pair at a time, with a straight needle, simply perforating each printed target from above each time, i.e. the needle was not passed underneath the surface of the target. Each trial consisted of six pairs, i.e. twelve total piercings. For this task, the performance was evaluated by time of completion, accuracy of the piercing, as well as number of mistakes (e.g. multiple perforations of the paper).
The second task was a pegboard transfer task based on the laparoscopic skills training curriculum from the Fundamentals of Laparoscopic Surgery program. This psychomotor task has been validated as a standard test of a surgeon’s efficiency and precision in a laparoscopic environment [63]. The user was asked to pick up rubber O-rings from a bin and place them on top of pegs (see Figure 6.2). Again this platform was 3D printed from lightweight plastic. Various sized rings and pegs were available, but each user was instructed to complete the same configuration of ring/peg pairings. The user was asked to place two small rings (internal diameter 3 mm) on the smallest pegs (outer diameter of 1.5 mm), and four large rings (internal diameter 6 mm) on the largest pegs (outer diameter of 3 mm). Thus this task had a slightly larger margin for error. User performance for this task was measured by time of completion and number of errors (e.g. dropped rings, missed pegs).

6.1.4 Results and Discussion

In both tasks, the user was evaluated in terms of their efficiency (measured in time), and their accuracy. For the suture test, the accuracy was determined by the distance of the needle piercing to the center of the printed dot; for the pegboard task, the accuracy was measured in the number of mistakes, i.e. how many times a peg was missed or a ring was dropped. The results are shown in Figure 6.3 and Figure 6.6.
Figure 6.2: Task 2: Peg and ring task

Figure 6.3: Results of Suture Task. Top-left: Average time of completion. Top-right: Average accuracy/distance from center of target. Bottom-left: Number of targets hit with an error of <0.5 mm. Bottom-right: Number of targets missed with an error of ≥1.0 mm
The suture task was the more technically challenging task for the participants, but the high precision required to complete the task made it a very good indicator of the effects of the various conditions on the participants’ performance. As can be seen in Figure 6.3, with the exception of user 2, the efficiency (performance time) deteriorated between conditions 1 and 2. The average accuracy also worsened, nearly doubling for most users when the target moved without compensation. However, once the compensation was enabled, the completion time was reduced slightly, and there was a significant improvement in accuracy. For most users the completion time with compensation enabled was slower than the stationary conditions, and the accuracy was also poorer. The effects of conditions 4 and 5, reduced amplitude of motion and reduced frequency of motion, showed varying results with the users. The users’ subjective evaluations, communicated in discussion after the tests, reflected these differences. Figure 6.4 shows the performance tradeoff between completion time and accuracy achieved for the various conditions. Generally a better accuracy (lower error) is obtained with longer completion times. The varying skill levels of the users is also demonstrated in this plot.

The bottom row of charts in Figure 6.3 shows a count of the average number of targets that were pierced within 0.5 mm as well as past 1.0 mm of the center of the target during the suture task. Each trial consists of piercing twelve targets. Figure 6.5 shows a summary of the results averaged across all the participants. These results demonstrate that there is a significant reduction in accuracy when the target is moving, however a lot of this accuracy is
Figure 6.5: Summary of inter-user results for accuracy and completion time for suture task

Figure 6.6: Results of Peg Task. Left: Average time of completion. Right: Average number of mistakes

recovered once compensation is enabled. Figure 6.5 also shows a summary of the change in completion time, relative to time to complete the stationary task, for each task. Comparing specifically conditions 2 and 3, the number of missed targets decreased from 37% to 13% when compensation was enabled, the number of high accuracy hits increased from 26% to 41%. The motion-compensated results are still not as successful as the stationary condition, but these are promising initial results.
The pegboard task was decidedly an easier task for users. This was reflected both in the quantitative results and the qualitative evaluations provided by the participants. Figure 6.6 shows a decrease in efficiency (increase in task completion time) between conditions 1 and 2, and a significant increase in the number of mistakes. With motion compensation enabled, the effect on time of completion and number of mistakes varied between users. It is expected that due to the low level of difficulty of this task, these results may be attributed to random factors and not necessarily the test conditions. Subjectively, all users felt that condition 3 was easier to complete. The reduction in amplitude and frequency seemed to have a more pronounced effect on the results in the pegboard task, showing inconsistent effects in the number of mistakes, but showing a decrease in completion time from condition 3. Despite the fact that a practice period was provided at the beginning of the test, it is expected these conclusions may be affected by natural human learning.

Further, it was observed in the user studies that an element of natural compensation was performed by the human user. Using the knowledge that the trajectory is periodic, the users were able to consciously predict the best location and best time at which to make a movement. This was clearly evident in the uncompensated trials, yet the results show the limits of what human compensation can achieve. With robotic compensation, the amount of human compensation that must be performed by the surgeon is reduced, and results in better accuracy and faster time of completion for these preliminary tests.

Statistical analysis using a two-tailed Student’s t-test was conducted, however the results did not yield statistical significance. This may be attributed to the small sample size in combination with the variance in their approaches to completing the tasks. Regardless, the results look promising enough to encourage future work. Two of the five users were expert robot users (robotic surgeons) for these experiments, and user studies with larger sample sizes will be conducted pending future hardware improvements.

We are aware that a number of limitations are present in our study. Our test setup requires further characterization and improvements. The heart trajectory that we use was determined from the da Vinci robot kinematics and has not yet been verified independently. We will carry out such characterization using an NDI Certus system in the near future. The camera used in the user studies uses the NTSC standard, and provides the surgeons with imaging resolution that is below what they are used to in their clinical systems. Further, we realized that our camera implementation for condition 2, with the camera fixed relative to the moving
platform, might be more difficult than a more realistic situation of a globally fixed camera. In our implementation only the instrument was moving relative to the camera, giving the illusion that the tool is in motion even though the surgeon’s hands are fixed; whereas with a fixed camera, the instrument would correctly appear stationary and only the target would be moving. Also, after data analysis, we also realized that the peg transfer task is not sufficiently challenging as implemented, but for future studies this could be easily adjusted to be more difficult. Finally, the most prevalent task in CABG surgery is anastomosis, but this was not included in our study.

6.2 Future Work

As a preliminary study in developing a clinically viable heart motion compensation system on the da Vinci surgical system, initial results and feedback from surgeons are promising. Future work will start by addressing the shortcomings mentioned in Section III.D. The peg transfer task will be made more challenging by increasing the peg size and reducing the ring size to leave a smaller margin for error. We will add a test condition similar to condition 2 but in which the camera is stationary. We will develop a similar tracking system for the da Vinci camera arm, which has different kinematics and dynamics than the manipulating PSMs. Once the camera can follow the heart trajectory, we will be able to study the effect of heart motion sensing delay (due, for example, to image processing for visual tissue tracking) on task completion and quality. Our testing conditions will be expanded accordingly. Anastomosis tasks will be added to our user studies with a suitable test of anastomosis quality and patency.

For clinical implementation, a heart-tracking system will require a closed loop controller, not an open-loop pre-compensator as used in our tests. We will start closed loop control by extending our spectral line approach, with the use of in-loop harmonic oscillators that provide infinite gain at the heartbeat spectral frequencies. We will consider adaptation and learning options to improve performance.

Understanding how users adapt to operating on a moving target is another interesting area of research that can be carried out by altering the tracking controller performance. In future user studies, human performance can be monitored while modifying the tracking delays or tracking errors, to determine what inaccuracies may be tolerated while still achieving the
required accuracy for anastomosis.

In parallel to controller development, work towards sensing heart wall motion robustly will continue. This area of research is an exciting visual servoing problem as three da Vinci arms would have to follow the heart motion in order to perform accurate surgery on the beating heart.

6.3 Conclusions

In this work we presented a novel application of the da Vinci surgical robot for heart motion compensation. We showed that the da Vinci Classic PSMs are capable of tracking an actual heart trajectory with small errors. To achieve these errors we used a simple approach based on the spectral decomposition of the desired trajectory. Simulating a virtually stabilized field, we evaluated the ability of surgeons to perform tasks on a moving, simulated heart target. All users reported the highest perceived difficulty for the condition of a moving target with no compensation. This was supported by slower completion times in both tasks, poorer accuracy and greater number of mistakes and additional perforations in the case of the suture task. When the motion compensation was enabled, the users felt a reduction in the difficulty of task completion. While their completion times were longer and had poorer accuracy than the gold standard of a stationary target, the loss in these measures was moderate. Improvement was observed between the cases of disabled and enabled compensation on the moving target, both objectively in terms of accuracy and completion time and subjectively in terms of perceived challenge. This improvement was achieved despite imperfections in the test setup and slight delays in the tracking. It is anticipated that with further development of the present system, performance comparable to the gold standard could be achieved for benchmark tasks. Further work is warranted to study anastomosis tasks under conditions similar to surgery.
This chapter will discuss an addition to the heart motion compensation system pertaining to including virtual fixtures, or active constraints. It was observed during the user study that minor latency errors were present in the system, and for this reason and perhaps due to the virtual stabilization, the user would accidentally perforate the heart surface as it was moving upwards towards the tool tip. As an added safety measure, the efficacy of a moving wall virtual fixture to prevent accidental collisions was evaluated. To reduce the amount of obstruction to the surgeon’s desired task, the virtual fixture is also designed to only be activated as the heart surface is accelerating upwards towards the surgical instrument, deemed the “high risk” portion of the trajectory. Instead of forbidding the surgeon from colliding with the heart surface (which is clinically infeasible; to complete the anastomosis the surgeon must penetrate the heart surface), this virtual fixture is meant to guide the surgeon to perform the majority of the instrument displacement during the “low risk” periods of the heart trajectory.

During a Visiting Student term at the Collaborative Haptics and Robotics in Medicine Laboratory in the Mechanical Engineering Department at Stanford University, this virtual fixture was developed and tested with user studies. The beauty of the dVRK, a shared medical research framework, is well demonstrated in this work since the heart motion compensation system from the University of British Columbia was ported over onto the Stanford system in a manner of hours on the software level. The results of the user studies were published in the proceedings for the 2015 Hamlyn Symposium on Medical Robotics [64]. The abstract follows in the remainder of this chapter.
7.1 Introduction

Virtual fixtures, or active constraints, are a controls concept that enable robots to intelligently collaborate with human users to complete tasks that are too complex for the robot to complete autonomously and too challenging for the user to complete unassisted. The body of work on dynamic active constraints is not extensive, but combinations of guidance and forbidden-region dynamic virtual fixtures for teleoperation have been studied [65]. Fixture evaluation methods in previous work include potential fields determined from pre-operative images, predicted and current positions of the environment surface location, and streaming point clouds. Some of these do not allow the slave to penetrate the forbidden region, which is impractical in surgical situations. This work proposes an impedance-controlled regional 3D dynamic virtual fixture generated based on dynamics of the heart motion. This virtual fixture has been added to the heart motion compensation system implemented on the dVRK described in [52]. In this work it was noted that while a user was operating on a virtually stabilized heart, the dynamics of the heart target were not accurately perceived. This resulted in additional, inadvertent perforations. To augment the surgeons perception, we propose a novel non-continuous, strobe-like virtual fixture whose enforcement is conditional upon the kinematics of a beating hearts trajectory.

7.2 Materials and Methods

7.2.1 Virtual Stabilization System Overview

The heart motion compensation system from [52] was implemented on the dVRK at the Collaborative Haptics and Robotics in Medicine Laboratory at Stanford University, seen in Figure 7.1. In the heart motion compensation system, the PSMs are programmed to follow the trajectory of a point on the heart surface, with superimposed teleoperation commands issued from the MTMs. This results in a virtual stabilization of the heart motion from the perspective of the surgeon. The 3D printed plastic heart target is controlled by one PSM while the surgeon controls the other PSM. The camera is attached to the plastic target at a fixed location relative to the simulated heart surface, emulating perfect camera motion tracking.
Figure 7.1: dVRK setup for virtual fixture experiments for beating heart applications at Stanford University.

Figure 7.2: High risk (blue) versus low risk (green) periods of the heart trajectory.
7.2.2 Virtual Fixture Design

In previous user studies for the heart motion compensation, it was difficult to see when the heart surface is accelerating upwards. This strobe fixture thus provides pulsing haptic feedback, active during periods of high acceleration in the heart trajectory. A 2nd order Butterworth Filter is used to calculate the z-acceleration, \( a_z \), of the heart using joint velocity readings from the position sensors in the PSMs. It was experimentally determined that, for the trajectory (as in [52]) with a z-displacement of 11 mm at 1.5 Hz, the virtual fixture strobe pulse commences when \( a_z > 0.39 \text{m/s}^2 \) and terminates when acceleration falls below \( a_z < -0.145 \text{m/s}^2 \), portrayed in Figure 7.2. The force feedback on the master is displayed only in the positive z direction. The magnitude of the feedback is calculated using the relative position of the slave tool tip and a proxy following the heart surface (similar to the concept in [66]). The tool tip and proxy follow an elastic linkage model:

\[
\delta_z = z_{\text{heart}} - z_{\text{tooltip}}
\]

\[
F_{\text{MTM}_z} = \begin{cases} 
V F_{\text{scale}} \times (\delta_z + V F_{\text{margin}}), & \text{if } \delta_z > 0, \\
0, & \text{otherwise.}
\end{cases}
\]

(7.2)

A virtual margin of 5 mm is added to the surface of the heart so the imperceptibly low \( F_{\text{MTM}_z} \) occurs before collision between the tool tip and the heart surface. The dynamics and inertia of the MTMs guarantee a smooth transition across the \( F_{\text{MTM}_z} = 0 \) threshold. \( V F_{\text{scale}} \approx 150; \delta_z \) is usually only a few millimetres so large scaling values are needed to generate perceptible haptic feedback. The torque applied to the MTM motors to display the haptic feedback is clipped to 1.0 Nm to avoid instabilities. The virtual fixture control loop runs at a frequency of 1 kHz.

7.3 User Studies

Seven right-handed subjects completed two one-handed tasks in an IRB-approved study with informed consent: drawing a circle (relevant to the standard Fundamentals of Laparoscopy circle cutting task), and a simulated suturing task (piercing printed targets with a
Heart motion compensation was active for both.

For drawing a circle, a 3D printed adapter was used to attach the marker coaxially to the instrument shaft. For suturing, a straight needle and 6 pairs of dots were used. Each dot is a circle 1 mm in diameter and the needle is 0.5 mm in diameter; high accuracy (similar to actual anastomosis) is required for this task. A target is “hit” if the needle perforation is completely within the printed dot, “missed” if the perforation is completely outside the printed dot, and “almost” otherwise. The subjects had varying levels of expertise with the robot and haptic devices. Each task was performed under four conditions; the subjects completed practice and three test trials for each. The conditions were:

- C1: no feedback
- C2: basic moving wall virtual fixture: feedback is calculated using the elastic linkage model (described in Section 7.2.2). This has been implemented in previous work [36], and is the baseline for comparing the effect of the non-continuity of the proposed virtual fixture
- C3: haptic pulse feedback: the pulse is active during high acceleration periods, irrespective of the relative positions of the tool tip and heart surface
- C4: pulsing moving wall virtual fixture: based off C2 with additional dependencies on the dynamics of the target
7.4 Results

The performance of the subjects under the various conditions was evaluated based on four indicators:

- Efficiency: time of completion
- Accuracy: how many times the marker tip left the surface (circle task); how many hit/missed/almost targets (suture task)
- Damage: the collision extent or time under the surface multiplied by the distance under the surface
- Safer Motion: to evaluate the efficacy of the non-continuity, we compare the motion of the teleoperated slave during the high-risk periods of the trajectory to the motion during lower-risk periods.

The results shown in Figure 7.4 (except for the circle results in row II) depict the average over the seven users. Efficiency and accuracy measures are normalized relative to the results for C1, showing the effect of the feedback.

7.5 Discussion

Based on the efficiency results, C2 was comparable to C4. However in the accuracy results for the circle task, C4 was the only condition with improved performance over C1. The collision extent results show the benefits of virtual fixtures, where all conditions with haptic feedback surpassed the results from C1. The ranking of the feedback conditions varied depending on the task. The advantage of the non-continuity is seen in the safe motion results, where in C3 and C4 the users seemed to perform a greater portion of the total displacement during the low-risk periods of the heart trajectory. There was high variance in performance between the different users; the tasks may have been too difficult for the skill levels of the participants, particularly the suturing task. Overall, the results show that the pulsing virtual fixture succeeds in relaying to the user the ideal times at which they should perform their tasks. The pulsing virtual fixture also improves completion time and accuracy of simpler tasks such as circle drawing.
Figure 7.4: Results for virtual fixture user studies. Left column shows circle task results, right column shows suture task results. Row I shows efficiency; row II shows accuracy; row III shows damage; and row IV shows safe motion results (green: low-risk periods, red: high-risk periods).
Chapter 8

Incorporating Motion Measurement

8.1 Introduction

While the results from the initial user studies on the proof of concept system described in Chapter 6 are very promising, the system must be developed further to progress from a bench-top proof of concept project to a clinically feasible system. There are a number of simplifying assumptions in the initial system, as well as shortcomings which are identified in Section 6.2. The next generation of the system, discussed in this chapter, incorporates an external device as the heart target, as well as an optical sensor to feed real time position data of the heart surface to the da Vinci controller.

We believe this is the first attempt to use external 3D sensing to command robotic manipulators for both visual and heart motion stabilization for bimanual teleoperation. In previous work, optical sensing was used to perform image processing to electronically stabilize the surgeon’s view of the surgical site [12]. In the initial system described in Chapter 6, the camera was mounted at a fixed distance from the heart target, and both the camera and the heart target were attached to a slave manipulator on the da Vinci. While this verified the concept of mechanical visual stabilization, it assumed a perfect motion measurement environment. The second generation of the heart motion compensate system eliminates this assumption and studies the effect of motion measurement errors for visual stabilization.

Further, since an external device is being used to simulate the heart, this relinquishes one PSM for other uses. For the study described herein, one PSM is used to carry the camera,
and the other two are used for bimanual teleoperation tasks. Initially the da Vinci Standard ECM was going to be used to move the camera, however the mechanical design of the ECM consists of ball screw actuation for the insertion joint. This results in the ECM being both too heavy and too slow to even hope to achieve the dynamics required to track the heart surface. Still, even with one PSM tasked as the camera arm, the ability to perform bimanual teleoperation tasks provides opportunities to study the system in the context of more realistic clinical tasks, particularly suturing.

8.2 Implementation

8.2.1 System Overview

The heart motion compensation system now consists of four computers, three responsible for controlling two robots and the 3D position sensor, and one with a frame grabber. A photograph and a diagram of the system are shown in Figure 8.1, Figure 8.2, and Figure 8.3.

![Photograph of the heart motion compensation system](image)

**Figure 8.1:** Photograph 1 of generation two of the heart motion compensation System
Figure 8.2: Photograph 2 of generation two of the heart motion compensation system

Figure 8.3: Block diagram of generation two of the Heart Motion Compensation system
respectively. The beating heart is simulated using a Novint Falcon haptic device, controlled by PC 1. The taskboard is mounted on top of the Novint in Figure 8.2. LED markers are affixed to the Novint Falcon, and are tracked by the (Northern Digital) NDI Certus Optotrak tracker. The data from the Optotrak is read via serial bus in PC 2. There is a TCP/IP interface between PC 2 and PC 3, over which PC 2 broadcasts the position of the Novint, as measured by the Optotrak. PC 3 streams in the Optotrak data, transforms it to be consistent with the dVRK coordinate system (using a coordinate transformation obtained through earlier calibration), and uses these positions to command the Cartesian displacement of the slave manipulators of the dVRK. In case the arm is also being teleoperated, the Cartesian displacements are superimposed upon the teleoperation commands. The subsequent subsections will discuss each component of the system in more detail. The fourth PC with a frame grabber records the surgeon’s view of the task.

8.2.2 The Heart: A Novint Falcon

Previously, a da Vinci PSM was used to carry a simulated heart target which was mounted to the shaft of an instrument. Issues with roll of the instrument shaft, using up a PSM, and having the heart target and heart motion compensation system on the same robot were all disadvantages of that idea, however it was useful in validating the proof of concept setup. To have a more clinically realistic setup, this generation of the heart motion compensation system uses an external device, the Novint Falcon, seen in Figure 8.4. While more anatomically realistic cardiac simulators are available, they are very expensive. A custom robot could have been designed and built to precisely recreate the 3D trajectory of the heart, however the Novint offered an affordable, simple solution to the heart simulation problem.

The Novint Falcon is a low-cost, 3D haptic device that is used for consumer and haptic applications. Offering 3D position and force feedback in a 4” × 4” × 4” workspace through
three parallel linkages, the Novint Falcon is often used for gaming and teleoperation applications. The Falcon drives its arms with three capstan drives and optical encoders for position sensing. The motor currents are commanded at a 1 kHz servo rate. Software developed by Novint, specifically the low level driver software called HDAL (Haptic Device Abstraction Layer), handles the serial communications between the Falcon and the controller computer. A C++ API is provided for HDAL, and this has been used to encapsulate the Falcon object in a Matlab Simulink s-function block, shown in Figure 8.5. Once in Simulink, a PID was tuned to control the Falcon to track the desired heart motion. Spectral line control, as in [52], was used for the Falcon, but this was contingent on not only errors in the Falcon but also in the calibration procedure, discussed later in Section 8.2.8.

Figure 8.5: Simulink control scheme for the Novint Falcon. An initialization phase is included, moving the Falcon to the desired start point with progressively more aggressive control.
8.2.3 The Sensor: NDI Certus Optotrak

The NDI Optotrak Certus® (seen in Figure 8.6) is a powerful research-grade motion capture system that features exception spatial and temporal accuracy. With its unsurpassed accuracy, it is used as the gold standard for motion tracking among research scientists [67]. The Certus boasts an accuracy of up to 0.1 mm and a resolution of 0.01 mm, performing real-time tracking of LED markers at frequencies up to 4600 Hz. NDI also provides an API, the Optotrak Application Programmer’s Interface (OAPI) which allows custom-written C++ code to interface with the Optotrak sensor information.

Using this API, I implemented software which initializes the Optotrak, forms a TCP/IP socket connection with the PC running the dVRK (the server is created on the dVRK PC, the Optotrak PC connects as the client), and streams the Optotrak continuously to the dVRK. There is unidirectional traffic across this socket, with data being written by the Optotrak PC and read by the dVRK PC. The socket is designed such that the dVRK PC always reads in the most recent position data. To be specific, the data being streamed from the Optotrak PC is the centroid of the 3D positions of the four markers arranged symmetrically around a point of interest, seen in Figure 8.7.
8.2.4 The Robot: the dVRK

The dVRK system has been described in previous chapters, though for the first time in this work all manipulators available (with the exception of the bulky ECM arm) are included in the system. The user may perform bimanual teleoperation tasks, with both the left and right master manipulators. This controls two slave arms, while the third slave is used to hold and position the camera (the same one as in the initial user studies is used here, and may be seen in Figure 5.12).

On the software side, a dispatcher component was created whose sole purpose is reading information from the NDI data stream over the TCP/IP socket and broadcasting it to the other components, as requested. The dispatcher opens the server on the dVRK computer at a specified port, and the NDI computer connects to this as a client. Once the dVRK code sees that the connection has been established, it starts reading from the socket at a frequency of 1 kHz. A timing analysis showed that new commanded positions were being supplied at a mean frequency of 333 Hz, which coincides with the loop frequency of the kinematics of the robot which is also 333 Hz. In cases where a new position was not received in time, the previous position was used.

The heart motion is not superimposed upon the slave arms until the user presses the “Start” button in the user interface, shown in Figure 8.8. This should not be initialized until the Novint Falcon target is stationary in its zeroed position, because the dispatcher...
component collects the first 100 ms of data to compute the origin of the streamed Novint position. The dispatcher then broadcasts to the relevant slave manipulators the displacement of the heart target, relative to that origin.

### 8.2.5 Calibration

The measured displacement of the Novint must be commanded to the dVRK in terms of the PSM coordinate frames. Thus, a calibration must be performed to find the trans-
formation between the NDI coordinate frame and the dVRK PSM coordinate frame. A semi-automated calibration procedure was developed to find $T_{NDI\rightarrow dVRK}$, or the $4\times4$ transform that transforms positions measured with the NDI to the dVRK coordinate frame.

The first step in the procedure is to command the dVRK PSM to travel a sinusoidal trajectory in the x, y, and z directions. The NDI marker plate from Figure 8.7 is mounted on the instrument shaft while the PSM performs this trajectory. The sinusoid has a peak to peak distance of 40 mm, and moves slowly enough the maximum/minimum points will be captured by the NDI. Five periods of the sinusoid are performed in each dimension, with a zero position commanded for the other two dimensions. A plot of the calibration trajectory is seen in Figure 8.9. The PSM Cartesian positions as measured by the forward kinematics with the encoders in the joints of the manipulator, as well as the measured centroid position of the NDI markers are both recorded.

Next, an automated algorithm processes the recorded data. The algorithm first partitions the logged data into three separate datasets, each one corresponding to the five sinusoids in one dimension. Next, iterating through the datasets, the algorithm extracts the maximum/minimum points in each period in the two measured trajectories (one measured by the sensors in dVRK and one by the NDI), which are then normalized and correspond to $\pm \hat{i}, \pm \hat{j}, \pm \hat{k}$, or the three mutually perpendicular unit vectors of the basis of each of the two Cartesian coordinate systems.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure8.9}
\caption{Automated calibration trajectory for the dVRK}
\end{figure}
At this stage the problem is a registration of two corresponding point sets. Also known as Wahba’s problem, this optimization problem seeks to find the optimal rotation and translation between corresponding 3D points \([68]\). Starting with point sets \(D\), of the dVRK points, and \(N\), of the NDI points. The problem becomes solving for

\[
D = RN + t
\]

where \(R\) is the \(3 \times 3\) rotation matrix, and \(t\) is the \(3 \times 1\) translation vector for the rigid transformation between the two point sets. So the transform, \(T\), is defined as

\[
T = \begin{bmatrix}
R_{00} & R_{01} & R_{02} & t_0 \\
R_{10} & R_{11} & R_{12} & t_1 \\
R_{20} & R_{21} & R_{22} & t_2 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

(8.2)

Letting \(P_D\) and \(P_N\) being the 3D point sets for the dVRK and the NDI, respectively, each with \(n\) points, the centroid of \(P_D\), \(c_D\), and the centroid of \(P_N\), \(c_N\) may be found by

\[
c_D = \frac{1}{n} \sum_{i=1}^{n} P_D^i
\]

(8.3)

\[
c_N = \frac{1}{n} \sum_{i=1}^{n} P_N^i
\]

(8.4)

The covariance matrix, \(H\), may then be obtained using

\[
H = \sum_{i=1}^{n} (P_D^i - c_D)(P_N^i - c_N)^T
\]

(8.5)

The optimal rotation, \(R\) is then found using Singular Value Decomposition (SVD),

\[
[U, S, V] = SVD(H)
\]

\[
R = UV^T
\]

(8.6)

(8.7)

and the translation, \(t\) is found using

\[
t = -Rc_D + c_N.
\]

(8.8)
This $4 \times 4$ transform is saved as a .txt file, which is read in through the GUI before the heart motion compensation is enabled. Since the origin is calculated using the first 100ms of data, as described above, the translation component of this transform is not really used. Instead, the $3 \times 3$ rotation matrix is critical in converting the NDI measurements to the dVRK space. This rotation matrix is stored in the dispatcher component, and before positions are broadcast to the slave arms, the measurement is transformed by the dispatcher to the dVRK space.

The NDI data collected during the calibration procedure is transformed using the computed $T_{\text{NDI} \to \text{dVRK}}$ and compared to the actual dVRK data. To find the angular error, the six principles vectors, $\pm \hat{i}, \pm \hat{j}, \pm \hat{k}$, from the dVRK data and the transformed NDI data are compared in corresponding pairs, e.g. $[+\hat{i}_{\text{dVRK}}, +\hat{i}_{\text{NDI}}], [-\hat{j}_{\text{dVRK}}, -\hat{j}_{\text{NDI}}]$. The maximum direction cosine error between each pair is computed. The maximum angular error over all the six pairs is computed to be 0.62 degrees, which corresponds to 0.12 millimetres over the 11 millimetre range of the heart motion in $z$.

### 8.2.6 Dealing with Delay

As a mechanical system with mass and inertia, there is an inherent delay of actually moving the manipulator to a desired position. With its cable drive mechanism the delay is small, however in the context of heart motion, where the heart moves over 10 mm about every second, a delay of tens of milliseconds can have a considerably detrimental effect. In the initial study, where the heart target was also being moved by a da Vinci manipulator, this inherent mechanical delay was consistent between the “heart” and the manipulator compensating for the motion, so this was not a huge issue. Now, with an external robot simulating the heart, the mechanical delay of the PSM is a large problem. System identification tests concluded that a delay of 55 milliseconds occurs between commanding the manipulator to a position and it actually achieving the position. A portion of this may be attributed to the mechanical delay, though the intrinsic delay in the joint level PID controller is also included in this.

To compensate for the delay, the assumed periodicity of the heart beat is leveraged. The length of the delay, $t_{\text{delay}}$ in milliseconds is known, as is the period of the heart beat, $T_{\text{heart}}$. Therefore, to align the manipulator’s achieved position with the measured position of the heart, the controller holds the desired reference signal $r(t)$ for $T_{\text{heart}} - t_{\text{delay}}$ milliseconds before sending it to the plant. This is displayed pictorially in the block diagram in Fig-
Figure 8.10: Block diagram of compensating for the system delay

In terms of software implementation, a First In First Out (FIFO) data structure is utilized: the queue. When a new position is received from the NDI, $r(t)$, it is pushed to the back of the queue, or enqueued. At 1 kHz the controller checks to see if the entry at the front of the queue has been held long enough, and if so, pops it from the front, or dequeues it, and commands the position to the arms, $r'(t)$. The resultant position signals are shown in Figure 8.11 and their errors in Figure 8.12. A significant decrease in error may be seen with the inclusion of the delay compensation, particularly in the z direction.

Figure 8.11: Received, $r(t)$, commanded, $r'(t)$, and measured, $y(t)$, position signals for the dVRK. The $r'(t)$ signal is the held $r(t)$ signal. $y(t)$ is the measured position of the dVRK PSM. Also, $r(t)$ corresponds to the measured position of the simulated heart target
Figure 8.12: Tracking error between heart target and dVRK position with and without delay compensation. The measured heart position is \( r(t) \), the dVRK position is \( y(t) \). \( r'(t) \) is the command signal with delay compensation.

8.2.7 Spectral Line Controller

The details of this controller have been explained previously in Section 5.4 and will not be repeated here. The implementation of the controller differs for this generation of the system, however, because the desired position signal for the dVRK is obtained from the transformed measured position of the Novint Falcon. Thus, the spectral line controller can be used on the Novint Falcon to try to optimize the trajectory being measured by the NDI and transformed into dVRK space.

Figure 8.13: Plant for system identification in spectral line controller development
In this case, the plant being inverted describes the Novint Falcon controlled by its PD controller, the measurement model of the NDI, as well as the coordinate transformation to the dVRK space, as seen in Figure 8.13. The same system identification process as was used, and the resultant bode plots for the x, y, and z axes are shown in Figure 8.14. With the aim of trying to have the da Vinci PSMs follow a trajectory that is similar to a realistic heart trajectory, demonstrated in the initial user studies, the new approach is built upon the initial system. Previously, a spectral line controller was used to compute a pre-conditioned input for the dVRK which resulted in sub-millimetre tracking error. Since the dVRK system has not changed, the goal is to have the added modules try to maintain the same input to the dVRK as used previously. Thus, the pre-conditioned dVRK input was defined as the desired output of the Novint-NDI-Transform modules, and a spectral line controller was used to compute a pre-conditioned input for the Novint. Figure 8.15 shows the outputs of the Novint-NDI-Transform system, or the input to the dVRK, \( r(t) \), when the pre-conditioned Novint input is used (blue), and when the original heart trajectory is used as the Novint input (red). These are shown in comparison to the original heart trajectory (green). Recall that even in the initial studies, the actual heart trajectory is not used as input to the dVRK. It is only shown in Figure 8.15 for reference. It can be seen that when the spectral line controller is used, higher frequency components which were interleaved in the position signal, introduced by noise and errors in the Novint, the NDI sensor and the coordinate transformation, are reduced. A discussion of final tracking error is discussed in
Figure 8.15: Comparing the input to the dVRK when spectral line control is used on the Novint (blue), and when just the original heart trajectory is commanded to the Novint (red), with the actual heart trajectory (green). Deviations from the actual heart trajectory only indicate how well the heart simulator emulates a realistic heart motion.

8.2.8 System Performance

When evaluating the performance of the system, the tracking error between the desired and achieved signals is the metric to be considered. Specifically, the difference between the heart target (Novint) position and the dVRK PSM position. There are four major components to consider when discussing this tracking error:

1. the Novint’s positioning accuracy (using its optical encoders and its proprietary kinematic model)
2. the accuracy of the NDI measurements of the markers rigidly attached to the Novint
3. the accuracy of the coordinate transformation from the NDI Cartesian coordinate system to the dVRK Cartesian coordinate system
4. the dVRK PSM positioning accuracy

These components are denoted in colour in the full system block diagram in Figure 8.3. The NDI and coordinate transformation may be conceptually abstracted into one “motion measurement” component. While each component boasts sub-millimetre accuracy, as described
in their respective sections, the cascading effect of these errors makes it very challenging to achieve sub-millimetre accuracy between the heart target and the dVRK PSM position. Even with finely tuned PID gains, delay compensation and the spectral line controller, to improve the overall tracking error to be sub-millimetre will require improvements to the individual components. The system and framework presented here is nonetheless a promising foundation upon which to build. Further, due to the introduction of these errors incurred from a realistic, imperfect motion measurement system, a heart motion frequency of 0.75 Hz was used. Slightly slower than for the initial studies, such a frequency could be clinically achieved using beta-blockers. The 1.5 Hz was an overestimate to begin with; discussions with clinicians say 1.0 Hz would be ideal for tests. For preliminary tests of the closed loop motion compensation system, a slightly slower frequency will be used.

In addition to the actual heart trajectory, \( h(t) \) there are five different signals of interest: \( r_N(t), y_N(t), y_{N,NDI}(t), y_{N,dVRK}(t) = r(t), \) and \( y(t) \), all denoted in Figure 8.3. The following figures display the tracking error between various signals, and for the sake of clarity they are depicted in separate figures. The first two figures show the tracking error of the first component: the Novint Falcon, between the preconditioned input commanded to the Novint and its achieved position, as reported by its own position sensors. This is simply a representation how well the heart simulator emulates the heart motion. The next two figures demonstrate the errors in the motion measurement component of the system (which includes NDI sensor and coordinate transformation). The final two display the tracking error of the dVRK PSM. Table 8.1 summarizes the maximum tracking errors in each dimension for each of these three modules.

Table 8.1: Maximum Tracking Errors for Closed Loop Heart Motion Compensation System

<table>
<thead>
<tr>
<th></th>
<th>X [mm]</th>
<th>Y [mm]</th>
<th>Z [mm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Novint Falcon</td>
<td>0.2697</td>
<td>0.5545</td>
<td>2.4915</td>
</tr>
<tr>
<td>Motion Measurement</td>
<td>0.3426</td>
<td>0.3947</td>
<td>1.1778</td>
</tr>
<tr>
<td>dVRK PSM</td>
<td>0.3636</td>
<td>0.9688</td>
<td>0.6944</td>
</tr>
</tbody>
</table>
Figure 8.16: Commanded versus achieved position signals for the Novint Falcon

Figure 8.17: Tracking error between commanded and achieved position signals for the Novint Falcon
Figure 8.18: Differences in position signals in the motion measurement component. $h(t)$ is the commanded heart trajectory, $T_{\text{NDI to dVRK}} \cdot y_N(t)$ shows the Novint self-reported position signal transformed into the dVRK space, and $y_{N,dVRK}$ shows the NDI measurements of the Novint transformed into the dVRK space (equal to $T_{\text{NDI to dVRK}} \cdot y_{N,NDI}(t)$). This shows the difference between where the Novint thinks it is moving, and where the NDI sees it move.

Figure 8.19: Error between NDI measured position and Novint self-reported position, both transformed into the dVRK space (i.e. $T_{\text{NDI to dVRK}} \cdot y_N(t) - y_{N,dVRK}$)
Figure 8.20: Commanded versus achieved position signals for the dVRK PSM. The commanded signal, \( r(t) \), is from the NDI measurements (transformed into dVRK space) of the Novint being commanded with its pre-conditioned input.

Figure 8.21: Tracking error between commanded and achieved position signals for the dVRK PSM.
8.2.9 External Validation

Having found the transform through the calibration procedure described in Section 8.2.5, external validation of the dVRK motion and the Novint Falcon can be performed. Each robot was commanded to follow a heart trajectory, and the NDI markers were mounted on each to record its motion with the NDI. The measurements from the NDI optical sensor were transformed into the Novint or dVRK space, as required, and compared with each robot’s self-reported position measurement as they were commanded to follow a heart trajectory. The validation for the Novint may be found in Figure 8.19, and in Figure 8.22 for the dVRK. In both, the commanded heart trajectory is shown in green, the transformed NDI measurements are shown in blue, and the position measurements reported by each robot are shown in red. These plots confirm that the dVRK and Novint are capable of tracking a realistic heart trajectory with reasonable error.
8.3 User Studies

8.3.1 Setup

A photograph of the full system setup was shown in Figure 8.1 and Figure 8.2, and an image of an expert user performing a test is seen in Figure 8.23. For the user studies, the Novint was positioned on the da Vinci patient side cart in line of sight of the NDI sensor, with the NDI markers rigidly mounted. Calibration between the Novint, NDI and dVRK is performed before every user study. The PCs for all these devices are positioned so they can all be operated during the user study. A fourth PC contains a frame grabber which records video data of the surgeon’s view during the study, for post-processing.

The three dVRK PSMs are positioned over the Novint; the center arm (green/PSM2) holds the camera while the right and left slave arms (yellow/PSM1 and red/PSM3) are positioned to enter the surgical site at about a 45 degree angle. The user sits at the master console and operates the right and left MTMs as if in a normal clinical setting. A Large Needle Driver was used in the teleoperated arms. The camera used was the same as in the initial studies and was held vertically over the surgical scene. The camera was repositioned by moving the setup joints of PSM2 during the test as requested by the user to get a better view of their task space. Three expert robotic surgeons with many years of experience with robotic suturing, all right-handed, participated in the study. Novice robot users were not invited due to the difficulty of the robotic suturing task; it was necessary the participants had previous expertise in suturing to evaluate the benefit of the motion compensation system.

8.3.2 Task Design

8.3.2.1 Task Target

The target application of this system is anastomosis for CABG surgery. The test task was therefore to perform bimanual suturing on a phantom under various conditions. An image of the phantom is seen in Figure 8.24; it is essentially a piece of dense foam coated in a coloured latex coating. 2-0 Vicryl sutures were used, and participants were asked to complete a double throw then single throw suture knot. An image of the procedure is shown in Figure 8.25. The user begins by wrapping the suture thread twice around the instrument
then reaches and grabs the loose end of the suture. After pulling tightly, another single loop is made around the instrument and again the loose end is pulled through. Both sides are pulled tight to secure the knot. The entry point for the suture was specified by a coloured dot on the phantom.

8.3.2.2 Test Conditions

The participants completed three practice sutures and five test sutures for each condition. A completed test target is seen in Figure 8.26. The conditions are

- C1: Everything stationary - gold standard; comparable to working on a stopped heart
• C2: Moving target, no compensation - emulates working on a beating heart without any aid from the robot

• C3: Moving target, with compensation - the condition that will validate this system; looks at performing tasks on the beating heart with robotic compensation enabled.

8.3.2.3 Metrics

Three objective and two subjective measures were used to evaluate the participants’ performance. The objective measures (and their units of measure) are

• Time of completion [sec]

• Total displacement of the MTMs [m]

• Accuracy of entry point [mm]

• Number of errors.

Figure 8.24: User study task phantom
Figure 8.25: Steps for tying the suture knot, images taken during user studies. Top-left: double throw around instrument. Top-right: Grab loose end. Bottom-left: Pull tightly. Bottom-right: make single loop and grab loose end again

Figure 8.26: Completed task phantom
The accuracy of the exit point would have been a desirable metric as well, however the needle size and tissue phantom properties were not very realistic, and would have skewed the performance results for the exit location. The subjective metrics aim to evaluate the ease with which the user could complete the task under the various conditions. The subjective measures are

- Perceived ease
- Perceived strain.

These were self-reported by the participants using a Likert scale after all the conditions were completed.

### 8.4 Results and Discussions

#### 8.4.1 Results

The following figures depict the results for the objective metrics. For the time data, the time for each trial was normalized relative to the mean of the five trials in condition one, the gold standard condition, for each user. The box plot in Figure 8.27 shows the results over all fifteen trials, five per user, for each condition. There was a significant amount of variance in the results, as can be seen in the whiskers of the plot. This variance stems from skill level of the surgeon, and external factors which could not be normalized. For example, on occasion, the throw of the suture would slip off the instrument. This was unrelated to the condition since it has no interaction with the tissue, the looping of the suture (steps a to d in Figure 8.25) is done in mid-air. This would cost the surgeon time. Figure 8.28 shows a scatter plot of the average normalized time for each condition, per user. The benefit of the motion compensation seems to depend on the skill level of the user, but this would be better understood with more participants. In Figure 8.28 user 2 was the least experienced surgeon. Figure 8.29 shows the sum of displacement of, or distance travelled by the MTMs during the suture tying. Again the data has been normalized relative the gold standard performance. We were interested to see if the user’s ease with a task was related to how much they moved their hands while completing it. An indirect measure of efficiency, if the user was more hesitant or slower in completing a task this would be reflected here. It appears the same
random factors – e.g. suture thread slipping off the instrument – which affected the timing results also affected the displacement results. The variance in the results is large, though the figure shows the median total displacement (sum of left and right motion) decreases marginally when motion compensation is enabled. Human learning may also be a factor affecting these results; there was a correlation between median displacement and how many sutures have been completed in the study.

The accuracy results were initially going to be measured in millimetres, however it was discovered that when the user punctured the foam at the indicated location with the needle and then tied the suture thread tight, the foam deformed and it was very difficult to get an accurate measurement of where the needle actually punctured the target. From watching the recorded videos, the success of each target could be classified as either “hit”, “missed” or “other”. A target was considered hit if the needle perforation was completely within the specified target, missed if completely outside, and other if it was touching the edge of the target. Recall each user was asked to tie five sutures per condition. The number of hit targets is shown in Figure 8.30. While not as good as the gold standard results, the compensation is shown to have an advantage over the non-compensated condition.

Figure 8.27: Suturing results: Time of completion box plot. The red line represents the median of the data, the blue edges of the box show the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers (which are depicted with a red cross)
Figure 8.28: Suturing results: Time of completion scatter plot.

The number of errors counts the number of additional perforations of the heart surface as well as missed grabs of the suture/needle due to it moving away on the heart surface. The results in Figure 8.31 show the mean number of errors per trial for each condition. An

Figure 8.29: Suturing results: MTM displacement. MTML: Left Master Tool Manipulator, MTMR: Right Master Tool Manipulator
observable decrease in the number of errors performed by the users is experienced when motion compensation is enabled, uniformly across all the users. This suggests that the motion compensation system decreases the risk of damaging the heart surface.

Finally, the last figure shows the results from the Likert scale, measuring self-reported perceived ease and strain for each user. The Likert scale included five options, and each option was assigned an integer value: strongly agree = 1, agree = 2, neutral = 3, disagree = 4, strongly disagree = 5. Thus, a lower score indicates more comfort, more ease, and generally a higher subjective preference for a condition. The results in Figure 8.32 once again show the difference in skill level and preference between the users. The average results show a tendency towards a higher preference for the motion compensated condition, those still not as much as for the gold standard condition. In conversation with the users after their tests, they mentioned that the residual motion of the heart in the motion compensated condition (due to the the millimetre-level tracking errors) was almost as distracting as the full heart motion. They could recognize the amplitude of the motion was significantly less when compensation was enabled, but the fact that it was still moving, even slightly, made it more difficult than the fully stationary condition.

![Suturing results: Accuracy](image)

**Figure 8.30:** Suturing results: Accuracy
Figure 8.31: Suturing results: Number of errors

Figure 8.32: Suturing results: Likert scale. Left: task was easy. Right: task induced strain. Strongly agree = 1, agree = 2, neutral = 3, disagree = 4, strongly disagree = 5. A lower score thus indicates more comfort
8.4.2 Discussion

The most obvious consideration is the sample size used in this analysis. Ideally upwards of ten participants could be included, and they would perform twenty sutures in each condition. This would allow the effects of uncontrollable factors such as suture thread slipping off instruments to be mitigated. Further, having more participants would allow a better investigation of the relationship between years of experience and performance with the heart motion compensation system. Unfortunately, the number of users in the Vancouver area who have the necessary qualifications to test this system are few in number. For this study all the known robotic experts in the area were invited, as were their residents who had sufficient robotic training. These individuals have very limited availability, so if they do have an opportunity to come test the system, the experiment cannot occupy hours and hours of their time. Giving an introduction to the experiment, performing the fifteen test sutures plus nine total practice sutures, and then completing the Likert questionnaire took approximately one hour of each participant’s time. Regardless, the results from this study have been very useful for learning critical lessons about studying surgeon’s ability to perform sutures.

There are many factors contributing to performing a clean, successful suture. The tissue type, suture thread, needle size, and type of knot are all highly important. Initially test targets were made of medium-density foam covered in a thick (about 2 mm) latex coating. The coating was found to be so stiff the experts were having tremendous difficulty puncturing it with the needle. The alternative targets, made of dense foam with a light latex coating, were thus employed for these tests. While the users were able to actually guide the needle through the tissue, all the participants noted that the tissue stiffness was significantly higher than anything they would deal with clinically, working in soft tissue. The suture thread used is also important; initially Monocryl thread was used, however this was so slippery the experts could not make the knot remain tightly tied. Instead, Vicryl thread was used in these studies. The needle size made less of an impact on the performance of the suture, though the size can help improve the clinical likeness of the test. Here, 2-0 sutures with SH 26 mm 1/2c taper needles were used, whereas for the anastomosis required in CABG surgery needles half the size would typically be used. Finally, the type of knot performed will directly affect the overall quality of the suture. The type of thread and the properties of the tissue must be considered when deciding which knot to use. For this study participants suggested that a cinch knot would have resulted in tighter sutures on the phantom.
The participants’ perception of the tracking error for the motion compensated conditions was also highly interesting. The residual motion from the tracking error they found to be very distracting. More careful investigation should be performed to evaluate whether the errors in the motion stabilization in the teleoperated arms, or in the visual stabilization in the camera arm are more distracting to the user. Ideally both would be eliminated entirely, however practically the da Vinci and the measurement model have limitations. There are opportunities for improved tracking performance on the manipulators discussed in the next section to improve the motion stabilization. Considering electronic stabilization for the improvement of the visual stabilization would also be beneficial.

8.5 Conclusions and Future Work

The results for the second generation heart motion compensation system are promising for the first clinically feasible system with real time motion measurement. The bimanual suturing, a highly surgically relevant task, served as a good indicator of practical use of the system since it is a very realistic task which demands high accuracy and comfort with the system since it is typically performed many times during a surgical procedure. The results show an increase in accuracy, a reduction in the number of errors performed, and a subjective preference for having motion compensation enabled when working on a moving target. As the first attempt at a closed loop motion measurement system, the results are promising, though there are many improvements which can be made to move closer to clinical feasibility.

One of the first things to investigate, low-hanging fruit, so to speak, would be to study the performance of the system with the originally oriented heart trajectory. Recall that the heart trajectory was rotated such that the maximal variance was in the z-direction, corresponding to the insertion direction on the PSMs. This resulted in about a range of [0.8, 2.4, 10.8]mm in the x, y, and z directions, respectively. In the original trajectory, the range is [2.3, 4.7, 9.2]mm in the x, y, and z directions, respectively. The range in x for the rotated trajectory is smaller than the positioning accuracy that can be achieved by the overall system. Further, once the teleoperated slaves are put into the approximately 45 degree entry position for their task, the z-direction no longer corresponds to the insertion joint, rather more to the outer yaw, which carries more weight and has to overcome more inertial effects. Investigating the potential improvement in tracking results when the original trajectory would be highly
interesting, and would not require any software, electrical, or mechanical changes to the system. Simply the input to the Novint would have to be modified.

On the theoretical side, it would be highly advantageous to develop a closed loop controller for the da Vinci, one more advanced that the basic joint-level PID already in place. A controller which could adapt with information from previous trajectory, learning from its tracking errors from the nearly periodic command signal, could potentially improve the tracking performance. The concept of harmonic oscillators, a controller with a pole placed directly on the harmonic frequencies of the heart trajectory, theoretically achieving infinite gain at these frequencies, was also considered to further reduce tracking error, but has not yet been implemented due to time constraints. A state space formulation of such a controller could be made adaptive.

Some participants were complaining about the residual motion in the visual stabilization caused by the small tracking errors in the system. Looking into electronic stabilization, i.e. performing image processing to ensure the same spot on the target is in the same position in the viewing screen every frame would complete the illusion of visual stabilization. Issues of image processing latency must be considered carefully, though.

System level changes may also be investigated. While the Novint and the NDI were selected due to their availability, convenience and believed good performance, the system analysis shows their sub-millimetre accuracies may not be sufficient for this application. Considerations should be made to look into a more accurate sensing technology, one which could ideally achieve accuracy in the order of tens of microns. With the goal of eventually working on porcine models, perhaps a more tissue-like phantom and heart simulator could be developed which would smooth the transition between bench top and clinical testing of the system. Finally, allowing the surgeon the ability to move the camera themselves would be advantageous to improve the clinical feasibility of the system. During the user studies the experienced surgeons began pressing the camera-controlling foot pedals from habit; operating the camera zoom and positioning its view to the ideal position is almost as important to the ease and comfort of completing a task as teleoperating the actual surgical instruments. While the prototype camera used in these studies does not offer many configurable settings (there is no zoom available, and the auto-corrections on the image cannot be modified), now it is mounted on a PSM the participant could be allowed to position the PSM as desired. This may be a complex addition to the system to properly incorporate the camera view into
the kinematics of the teleoperation.
Chapter 9

Conclusion and Recommendations

9.1 Conclusions

This thesis work discusses the development of a heart motion compensation system implemented on a da Vinci® Research Kit controlling a clinical da Vinci Classic System, for the application of anastomosis during minimally invasive CABG surgery on a beating heart.

The preliminary research hypothesis of whether the robot was capable of tracking the heart motion is proven to be true by achieving tracking error on the order of one millimetre of a realistic 3D heart trajectory. This heart trajectory corresponded to a published 3D heart trajectory, measured on a point on the left ventricular midwall. The trajectory is assumed to be periodic with a frequency of 1.5 Hz with a maximum displacement of 11 mm. A spectral analysis of this heart trajectory was performed, and a spectral line controller was developed with the findings. System identification on the da Vinci slave arms was performed to obtain a model of the system, and using the principle of inversion at the heart spectral frequencies, the open-loop spectral line controller computed a pre-conditioned input which minimized the tracking errors.

With the slave arms capable of following the heart trajectory, the surgeon’s teleoperation commands were superimposed upon the automated trajectory. A stereo endoscopic camera was also commanded to follow the heart trajectory, emulating working on a stopped heart.
The secondary research question was whether the motion compensation system would be positively received by robotic surgeons, and how a semi-automated, shared control teleoperation scheme would compare to the gold standard of performing surgery on a stopped heart. Initial user studies with expert robotic surgeons showed very promising results, with significant improvement in accuracy from the uncompensated to compensated conditions. The accuracy and completion times were not as good as the gold standard results of working on a stopped target, however the surgeons were excited about the system and its potential implications for the field of robotic cardiac procedures.

The second generation of the system, incorporating real time motion measurement, presents a more clinically feasibly closed loop system. An external target (a commercially available robot) is used to simulate the beating heart, a 3D optical position sensor (NDI Certus Optotrak) measures the displacement of the heart, the measurements are transformed into the dVRK coordinate space (using a coordinate transformation determined during a calibration procedure), and then the dVRK slave manipulators are commanded to compensate for the heart motion. Delay compensation was integrated to handle the temporal delays totalling 55 milliseconds that are present in the system. The periodicity of the heart motion signal is used to accomplish this; information from the previous period is used to command the robot to minimize the tracking error. All three PSMs are used in the second generation system: one arm carries the small endoscopic camera from the initial studies (the assumption of perfect motion measurement is removed), and the other two arms are teleoperated by the surgeon. This setup allowed for bimanual suturing to be performed on a foam target during the user studies. A highly pertinent clinical task, the results showed the motion compensation improved the participants’ accuracy and reduced the number of errors performed. Subjectively, the users found themselves to be more strained when motion compensation was enabled, compared to the gold standard of a stationary target, due to the tracking errors and motion measurement errors.

As an additional safety feature, virtual fixtures were considered to attempt to reduce accidental perforations of the heart surface. With the virtual stabilization provided by the heart motion compensation system, as well as with the errors present in the system, it was sometimes difficult to perceive when the heart surface is accelerating upwards towards the instrument tool tip. As an aid for the surgeon, the virtual fixtures would provide haptic feedback during periods of acceleration upwards towards the instrument, indicating a “high risk” portion of the trajectory. This was a dynamic, non-continuous virtual fixture which
was also verified in user studies. Results showed the fixtures were helpful in relaying when the operator should be performing the majority of their hand motions.

Another contribution was the development of a Simulink to C++ interface for rapid prototyping of controller design. The controller design and development process in the context of the dVRK cisst/SAW framework was very tedious and time-consuming. This interface, based on TCP/IP communication, effectively outsources the control logic to Simulink, a friendly block-diagram environment which also offers many useful toolboxes for controller analysis and design. Experiments showed a 1 kHz control frequency could be maintained when the Simulink controller was commanding the robot. This will be a useful tool for control system designers on the dVRK.

An advantage of the work presented in this thesis is the employment of the da Vinci as the robotic platform for the heart motion compensation system. Already clinically prevalent, the da Vinci is poised as the perfect platform upon which novel automation schemes may be introduced into clinical use. With thousands of da Vinci systems already in use, the potential impact of a da Vinci based system may be highly significant. However, the work done here was completed on the first generation da Vinci system, and must be modified to work on the newest, substantially different system.

As a proof of concept development, this system still contains a number of limitations. The model of the manipulators used for control development were based on parametric models obtained using data-driven system identification. More accurate models may be obtained with nonparametric methods. The heart trajectory used for the developments was assumed to be periodic, when in reality the heart has a quasiperiodic motion signal. A closed loop controller will be necessary for clinical implementation, and vision based sensing may not be ideal since occlusions will be a significant concern in clinical application. Nonetheless, as a base system the results were very promising. Future improvements that can be made to address these concerns are addressed in the next section.

9.2 Contributions

To summarize, the contributions of this thesis work are as follows

- the development of an open loop spectral line controller which is proven to be ca-
pable of commanding the da Vinci patient side manipulators to track a realistic 3D heart trajectory with *submililetre tracking error*, assuming periodic quasiperiodic trajectories.

- a novel idea for *visual stabilization*, consisting of mechanically moving the camera to compensate for the heart motion, keeping a fixed distance between the heart surface and the camera.

- the development of additional safety measures to protect the heart surface, in the form of *non-continuous moving wall virtual fixtures*.

- the creation of a *controller development tool* linking Simulink to the C++ dVRK code, allowing the outsourcing of control logic to Simulink.

- incorporating a *real time, 3D optical position sensing* modality to command the da Vinci patient side manipulators to track the heart surface. Still using an open loop spectral line controller, the closed motion measurement system is now more clinically feasible. Tracking errors on the order of one millimetre were achieved for this system.

- developed a reasonable approach to human studies, evaluating the overall integrated system and controller with expert robotic surgeons to evaluate the system’s ability to help the surgeon work on a moving target. Clinically relevant, *complex bimanual teleoperation tasks* were completed by the surgeons in the form of suturing foam targets. The results from the studies show significant improvement between compensated and uncompensated conditions when working on the moving target. The system can be further improved by iterating the control scheme using the Simulink tool.

- demonstrated the feasibility of a *heart motion compensation system on the da Vinci system*, an FDA clinical system that has been used in thousands of surgeries.

### 9.3 Recommendations

A number of improvements and further advancements are necessary before the system can be considered for clinical use. The long term developments would hopefully lead to porcine model testing, followed by clinical studies, obtaining FDA approval and ultimately clinical practice. The most prominent is the incorporation of motion measurement of the
heart. A clinically feasible solution for acquiring the position signal corresponding to the heart surface trajectory will be challenging. Past research shows tissue tracking using stereo endoscopic cameras is possible, particularly when augmented with information from electrocardiogram signals. Further controller development to minimize latency and attempt to improve tracking errors with a closed loop controller would also be required.

More immediately, user studies with more surgically relevant tasks, such as an actual anastomosis including more complex articulated instrument motions and instrument/tissue interactions, would be a better indicator of the quality of the system. One of the expert surgeons who participated in the second generation user studies suggested either real heart tissue and a pump flowing blood through the vessels, or else obtaining some simulated vessels available from Ethicon, used in regular surgical training. Further, he recommended performing experiments with the original, non-rotated heart trajectory signal. The true difficulty of the task on a beating heart, without robotic compensation, is not well portrayed with so little motion in the x- and y- directions.

While there are many challenges still to face, many of these may be mitigated by simplifying the core motion compensation problem. Novel, minimally invasive mechanical stabilizers can be designed which would constrain the heart motion, minimizing the amplitude of the motion compensation required. Further, pacemakers and drugs such as beta blockers can be used to slow down and pace the heart. The lower frequency, lower amplitude motion will result in simplified trajectories for the robot to track, and hopefully even better tracking results.

The targeted application is minimally invasive CABG surgery on a beating heart, since this is the most common revascularization procedure, although this can also be utilized for other cardiac applications such as mitral valve repair. The motion compensation principles developed may also be employed for respiration compensation. It is acknowledged that not all patients will be eligible for minimally invasive surgery on the beating heart. Prior heart conditions, difficult access to the surgical site, and medical history might require a traditional open procedure. However, if clinically accepted for robotic cardiac procedures, this system may have a largely significant impact, particularly in popularizing robotic surgery in the cardiac space. Using the already prevalent da Vinci surgical system as the robotic platform significantly improves the likelihood of such a system being adopted into practice, since only the automation component on top of the system must be FDA approved. if
successful, the need for both the sternotomy and the CPB machine would be eliminated, and patients could potentially walk away from cardiac bypass surgeries the same day. This stands to not only greatly improve patient quality of life, but also significantly reduce hospital time and costs.
Bibliography


Appendix A

Running an Experiment

This appendix has been included as an instruction set to the next poor soul who is taking on this project and wants to repeat the experiments. These instructions are for the user studies described in Chapter 8, to test the second generation system. Before I start, my sincerest apologies, the system grew quickly in complexity near the end and I tried to keep it as neatly organized as possible. There may be better ways of doing it. Good luck!

1. The first step: become well acquainted with the robot. I mean it. Develop a solid relationship, get to know Dr. Jekyll and Mr. Hyde’s quirks and preferences. It’ll help you both in the long run. This machine is pushing 20 years and new parts are no longer being produced. Take care of it. Spend some time looking at the dVRK Software Overview available from the User Meetings on the Intuitive dVRK User Group Website. Try playing with simpler examples first, like mainQtArm.cpp, mainQtTeleOperation.cpp and mainQtFollowTrajectory.cpp to figure out what’s going on.

2. Deep breath, you’re ready to go

3. Turn all the devices on

   (a) Turn on the da Vinci. Make sure all five arms are plugged into their dVRK controllers. Power the controller boxes in the order of the firewire daisy chain, top to bottom. Wait 10 seconds for the FPGA and QLA boards to initialize then plug the firewire into the top of the daisy chain.

   (b) Turn on the NDI - this includes the large gray sensor as well as the sensor box. Make sure all the markers/strobers are plugged in as necessary.
(c) Turn on the Novint

i. Plug it into the `dragonfly.ece.ubc.ca` computer. This PC has the correct system requirements to run the Falcon.

ii. If heaven forbid you plug a new Falcon in, re-install the drivers. Windows tries to be smart and automatically installs/overwrites the drivers when you plug in a new device, but these don’t work. You’ll hear an annoying buzzing sound from the computer if you try to communicate with the Falcon. The executable to install the drivers is found in `C:\ProgramFiles\Novint`.

iii. Check to see that it’s working correctly with the Falcon Test utility that comes installed with the Falcon (there’s a link off the desktop), found at `C:\ProgramFiles\Novint\Falcon\TestUtilities`.

iv. Run a trajectory, the LED lights will shine red until you manually move the knob around to home the motors.

v. Simulink models for the Novint and running the heart trajectory are in `C:\Documents and Settings\Controller\Desktop\Heart motion trajectory Novint_PD_ctrl`.

(d) Turn on the da Vinci Master Console and camera controller box.

(e) Turn on all four computer involved: one for each of Novint (`dragonfly.ece.ubc.ca` or `DVRK SENSOR`), NDI (`purang1.ece.ubc.ca`), dVRK (`tims-pc12.ece.ubc.ca` or `DVRK PC1`), and frame grabber (`FRAME GRABBER PC`, it’s got a colourful collection of cables hanging off it).

4. Position the da Vinci slave arms by moving the setup joints as desired.

5. Start the dVRK software. Launch `mainQtTeleOperationVisualSevoing5.cpp` application with the correct parameters.

6. Find the optimal position for the phantom on the Novint on the patient side table. Make sure the camera can get a good view of it.

7. Perform calibration between the NDI and the two robots

   (a) Launch the NDI streamer on the NDI computer
       
       (C:\\NDIopapi\aruszkow\NDIOAPI_svn\vc\NDIStreaming.sln

   (b) Open the stream on the dVRK side through the GUI

   (c) Make sure Falcon markers are in place and plugged into the strober
(d) Prepare the Falcon to run the calibration trajectory (run `makeCalibTraj.m`, then set `traj = makeCalibTraj`.

(e) Launch the Falcon to follow its calibration trajectory. Once it is initialized before it starts the calibration trajectory, “Run Falcon Calib” to start collecting the streamed NDI data.

(f) Once the Falcon is done, stop both the Falcon and the NDI stream by hitting “Run Falcon Calib” again. A `calib_NDI_to_Falcon.log` should have been saved in your build directory.

(g) Run `saveData.m` and commit the updated `falconData.mat` file to SVN. Update the corresponding SVN folder on the dVRK PC to have the Matlab Simulink simulation data on the dVRK PC.

(h) Unplug the Falcon markers and plug the dVRK markers into the NDI strober. Make sure the dVRK markers are in place, centered over O6.

(i) Uncheck the teleoperated arms and make sure the correct calibration trajectory is selected for the camera arm. Ensure the z-offset is correct. Run the dVRK calibration.

(j) Press “Run dVRK calib” again once the trajectory is done. Two more log files should have been generated.

(k) Copy all pertinent log files (three named `calib_NDI_something` and the `falconData.mat`) to a desired location.

(l) Run the `mainAutoCalibration.m` script in Matlab.

(m) Rename the auto-saved T.txt in the `rcltrans` (see `.bash_aliases` on the dVRK PC) folder to something meaningful.

(n) Plug the Novint markers back into the strober and remove the dVRK marker mount.

8. Ready to start the actual test! Make sure the camera arm is well positioned and the offset is updated.

9. Select the teleoperation scaling factor to be consistent for both teleoperation modules (left and right).

10. Might be a good idea to see if all arms can follow a simple z-sinusoid trajectory to check everything is set up correctly.
11. Prepare the BlackMagic Express capture on the Frame Grabber PC

12. Make sure have no logs already (rm *.log in rclb)

13. For each condition

   (a) Start with fresh suture, cut to half length. Explain they need to do at least 8 sutures with it

   (b) Give user three practice sutures

   (c) Ask them to do five test sutures, all along one row on the target board. Give them a count down to start and enable logs in the dVRK GUI and start recording what they see in the frame grabber on “GO”. Make sure to stop recording and disable logs once they say they’re done.

14. Ask user to complete the Likert test

15. Can use the SVN Matlab scripts I developed for data analysis (ReadCustomLog.m is the best) as necessary, after the test.

16. Clean up and power down in the opposite order of setup.

17. All done, piece of cake! That wasn’t so bad :)}
Software Overview

First thing’s first, I must disclose I am a huge supporter of version control and was managing all the source code for the robot, as well as my personal code, using either GIT or SVN, through my Unfuddle account that’s managed by Tim.

B.1 SVN

I have asked for a repository for all my lab work and was granted: https://rcl.unfuddle.com/svn/rcl_angelicarl/. Inside that repository I have some of my course work, but more interestingly is the Thesis directory, which contains four subdirectories:

1. HeartTrajectoryCode
2. LogDataAnalysis
3. NDIcalibResults
4. NDIcalibration
5. NDIOAPI_svn

Most of my code is well commented, and I’m a fan of ridiculously long variable names, because I’d often forget myself what my code was supposed to do. Hopefully it’s not too bad to read. Look for my README files, as well. I left a bunch around to again remind myself what a particular script/test/file was for.
B.1.1  HeartTrajectoryCode

Contains Matlab scripts for performing PCA, calculating FFTs, performing reconstructions, and otherwise processing and manipulating the heart trajectory data.

B.1.2  LogDataAnalysis

Contains a whole myriad of Matlab scripts that do all sorts of things. I tried to give the scripts clever names to remind myself what they do. Generally these scripts parse log files produced by the dVRK software, plot results, find transfer functions, extract critical points, and all sorts of fun helper functions. Most important is the ReadCustomLog.m script which I have been solely using to read my .log files generated by the dVRK software. Easy to modify, just change a few variables at the beginning and you’re good to go.

B.1.3  NDICalibResults

This is where the data from the Matlab Simulink simulation on the Falcon computer gets uploaded. Could’ve used a USB stick, this was easier.

B.1.4  NDICalibration

Contains the development, helper function, and main file for performing the calibration between the Falcon and NDI, and NDI and dVRK to find the coordinate transforms between the three elements. Start with main_autoCalibration.m.

B.1.5  NDIOAPI_svn

This holds the source code for the Visual Studio project which uses the Optotrak API (OAPI) to stream the NDI measurements to the dVRK PC.

B.2  GIT

Fair warning: I’m going to assume here that the reader has basic experience with git.
The original open source CISST/SAW code is in a github repository found at https://github.com/jhu-cisst/cisst-saw. Since there is a substantial amount of code that runs the dVRK, and with multiple projects going on the lab I figured the best way to keep everything straight was just to divide all the code such that each project has its own git branch.

In the main JHU github repository there are a number of submodules, essentially pointers to other repositories, each containing a specific module. Pro tip: sawControllers and sawIntuitiveResearchKit are the one you’ll probably be modifying for your work. The branches are organized such that there is one master branch for the main repository, but 8 branches in each of the submodules. If you ever want to switch branches, go to the top directory and use the switchBranch.sh script. Although most submodules aren’t used, it’s cleaner to just have all the submodules switch together, instead of remembering which submodule is on which branch. The branches are

1. **master** (Basically JHU’s code, with the addition of my followTrajectory code)
2. **dante** (Co-op student’s surgeon training project)
3. **gaze** (Irene’s gaze tracking code)
4. **movingECM** (Omid and I tried to get the ECM to move, this is basically closed; JHU came out with better code)
5. **simulink** (Contains development and code for the Simulink controller project described in Section 5.3)
6. **virtualFixtures** (Contains code for my work described in Chapter 7)
7. **visualServo** (Contains code for the closed loop motion measurement system; basically the second generation of the heart motion compensation system)
8. **waypointMode** (Caitlin’s project to command the position of the dVRK through TCP/IP socket communication from an ultrasound machine, coordinating moving a pickup ultrasound probe with the dVRK and scanning the image with the ultrasound machine)

The main dVRK machine is set up to pull from the JHU repositories and push to RCL’s Unfuddle repositories. This should be the only PC connected to JHU’s repository. All other machines in the lab should deal exclusively with the RCL Unfuddle repositories. Instructions on how to do that are found here: https://intranet.rcl.ece.ubc.ca/wiki/AngelicaGitTips.
Actually, I used to keep a detailed log when I was struggling through learning the dVRK code, all on the RCL website, found here: https://intranet.rcl.ece.ubc.ca/wiki/AngelicaLog.