6-20-2017 12:00 AM

Design and Evaluation of a Contact-Free Interface for Minimally Invasive Robotics Assisted Surgery

Cameron Dawson
The University of Western Ontario

Supervisor
Ana Luisa Trejos
The University of Western Ontario Joint Supervisor
Rajnikant Patel
The University of Western Ontario

Graduate Program in Biomedical Engineering
A thesis submitted in partial fulfillment of the requirements for the degree in Master of Engineering Science
© Cameron Dawson 2017

Abstract

Robotics assisted minimally invasive surgery (RAMIS) is becoming increasingly more common for many surgical procedures. These minimally invasive techniques offer the benefit of reduced patient recovery time, mortality and scarring compared to traditional open surgery. Teleoperated procedures have the added advantage of increased visualization, and enhanced accuracy for the surgeon through tremor filtering and scaling down hand motions. There are however still limitations in these techniques preventing the widespread growth of the technology. In RAMIS, the surgeon is limited in their movement by the operating console or master device, and the cost of robotic surgery is often too high to justify for many procedures. Sterilization issues arise as well, as the surgeon must be in contact with the master device, preventing a smooth transition between traditional and robotic modes of surgery.

This thesis outlines the design and analysis of a novel method of interaction with the da Vinci Surgical Robot. Using the da Vinci Research Kit (dVRK), an open source research platform for the da Vinci robot, an interface was developed for controlling the robotic arms with the Leap Motion Controller. This small device uses infrared LEDs and two cameras to detect the 3D positions of the hand and fingers. This data from the hands are mapped to the da Vinci surgical tools in real time, providing the surgeon with an intuitive method of controlling the instruments. An analysis of the tracking workspace is provided, to give a solution to occlusion issues. Multiple sensors are fused together in order to increase the range of trackable motion over a single sensor. Additional work involves replacing the current viewing screen with a virtual reality (VR) headset (Oculus Rift), to provide the surgeon with a stereoscopic 3D view of the surgical site without the need for a large monitor. The headset also provides the user with a more intuitive and natural method of positioning the camera during surgery, using the natural motions of the head. The large master console of the da Vinci system has been replaced with an inexpensive vision based tracking system, and VR headset, allowing the surgeon to operate the da Vinci Surgical Robot with more natural movements for the user. A preliminary evaluation of the system is provided, with recommendations for future work.

Acknowledgements

First and foremost, I would like to thank my supervisors, Dr. Ana Luisa Trejos and Dr. Rajni Patel, for their continued guidance and support throughout my graduate studies and thesis project. Each has played a critical role in the development of this work, and have kept me on track throughout. The input from my advisory committee, Dr. Michael Naish, and Dr. Jim Lacefield, is also greatly appreciated.

I was privileged to be able to work with a number of talented individuals at CSTAR and Western University, and would like to thank everyone at CSTAR who has helped with this project in any way. I could not have completed this project without the generous help and suggestions from Abelardo Escoto and Christopher Ward from the beginning of my time here. They were a tremendous help in many aspects of the project, from answering any and all questions I had, to help with any equipment in the lab, and troubleshooting any problems that arose. I would like to thank Ran Xu as well for his suggestions into the project, as well as Karen Siroen for her help in organizing trials and receiving ethics approval. The entire team at CSTAR has made the past two years a wonderful experience.

I would like to acknowledge Paul Sheller and Stephen Mallinson at the Engineering Finance Stores for their help with processing orders. As well as Frank Van Sas from the Physics and Astronomy Department for machining and 3D printing of components for the project.

Finally, I would like to thank my parents, and my girlfriend Jennifer Van Sas for their support, and encouragement throughout my graduate studies. Without them, this all would not have been possible.

Contents

List of Figures

List of Tables

Nomenclature and Acronyms

Acronyms

2D	Two-Dimensional
3D	Three-Dimensional
AR	Augmented Reality
CAD	Computer Aided Design
CV	Computer Vision
DOF	Degrees of Freedom
dVRK	da Vinci Research Kit
ECM	Endoscope Control Manipulator
EM	Electromagnetic
HMI	Human Machine Interface
IMU	Inertial Measurement Unit
IPD	Inter-pupillary Distance
IR	Infrared
LED	Light-Emitting Diode
LCD	Liquid Crystal Display
MCP	Metocarpophalangeal
MTM	Master Tool Manipulator

OS	Operating System
PC	Personal Computer
PSM	Patient Side Manipulator
RAMIS	Robotics Assisted Minimally Invasive Surgery
RGB	Red, Green, and Blue
RMS	Root Mean Square
ROS	Robot Operating System
RPY	Roll-Pitch-Yaw angles
SDK	Software Development Kit
VR	Virtual Reality

Chapter 1

Introduction

Many traditional surgical procedures are very invasive, causing much pain and discomfort for patients during recovery. In open surgery, a large incision is made to access the organ of interest, causing trauma to the surrounding tissue, resulting in a long recovery time. To mitigate some of the negative effects of open surgery, minimally invasive surgery (MIS), or laparoscopy has been developed, which has revolutionized surgical care. The benefits of these procedures are less pain and blood loss, shorter hospital stay, improved cosmesis, and reduced morbidity rates [1–3]. In contrast to open surgery, where a large incision is made to directly view the organs of interest, MIS is performed to reduce the amount of trauma as much as possible. Small incisions are made in the skin, and long instruments inserted into the body cavity through a trocar, and remotely manipulated outside the body. The surgeon monitors the surgical site by use of an endoscopic camera inserted through one of the incisions. Figure 1.1 shows a comparison of MIS and open surgeries. Although the benefits of these surgical procedures have been shown, there are additional drawbacks that are introduced when utilizing these MIS methods. The tools are inserted though a trocar, thus limiting the degrees of freedom of the instrument, introducing a lack of dexterity as compared to open surgery. Due to the contraint of the trocar, the surgeon's hands motions are also reflected about the incision point, known as the "fulcrum effect" [4] (Figure 1.2). Lastly, standard 2D endoscopes used in laparoscopy remove the depth perception one has with binocular vision. These factors combined create a steep learning curve for surgeons in training [5–7].

Within the past 20 years, there has been extensive research into incorporating robotics in

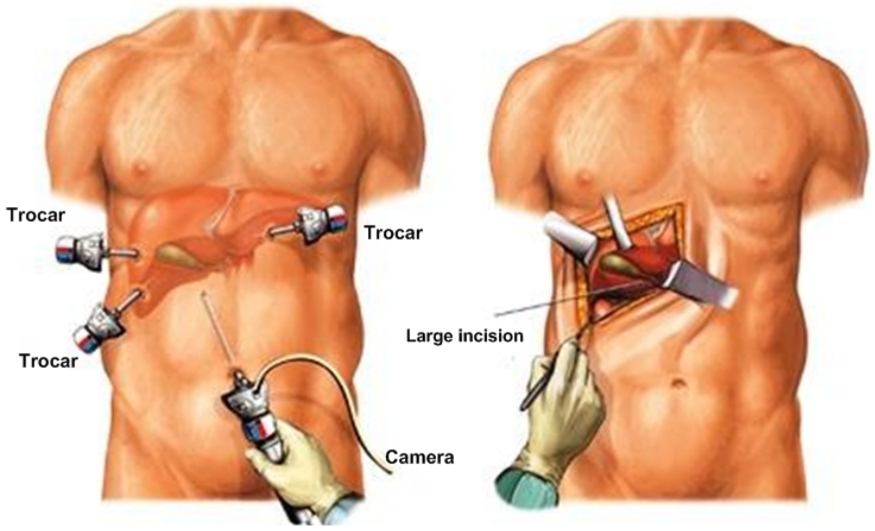


Figure 1.1: Laparoscopic vs open surgery (©gallbladder-help.com)

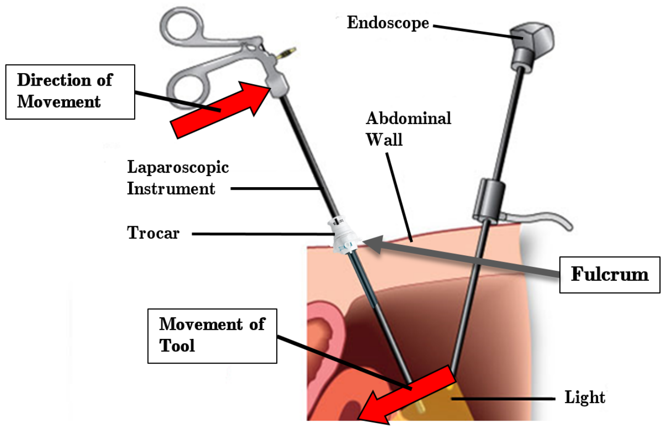


Figure 1.2: Laparoscopic tool showing fulcrum effect

surgery, and the advancement of robotics-assisted minimally invasive surgery (RAMIS) has been one of the largest breakthroughs in surgery since anesthesia [8]. The advantages that RAMIS brings, are the increased dexterity due to wristed instruments, scaling down hand movements for increased precision, and filtering tremors of the surgeons hands during surgery. The technology allows surgeons to operate in the same way they would in open surgery, without having to mirror the movements of their hands. The limitations of laparoscopic surgery previously mentioned have been improved greatly: the movements are no longer inverted, and dexterity and precision is increased, allowing for inreased surgical performance [9–11]. The improvements have shown to enable a reduced operation time, shorter hospital stay and quicker recovery time over traditional laparoscopy [12–14].

The most widely used RAMIS system is the da Vinci® Surgical System (Intuitive Surgical, Inc.) [15], the first system approved for clinical use. There have been other RAMIS systems introduced, but the da Vinci will be the main focus of the work in this thesis. The da Vinci is a master–slave robotic system, enabling the surgeon to perform surgical operations teleoperatively. Procedures performed with the system rely on the visual feedback given to the surgeon for navigation. The surgeon operates in a seated position at the surgeon console, and manipulates the controllers to manoeuvre the robotic arms at the patient cart, while viewing the surgical site through a binocular view finder. Figure 1.3 shows the surgeon console, and the patient side cart with surgical instruments. Figure 1.4 shows the interface the surgeon operates with, including the operative screen for visualization.

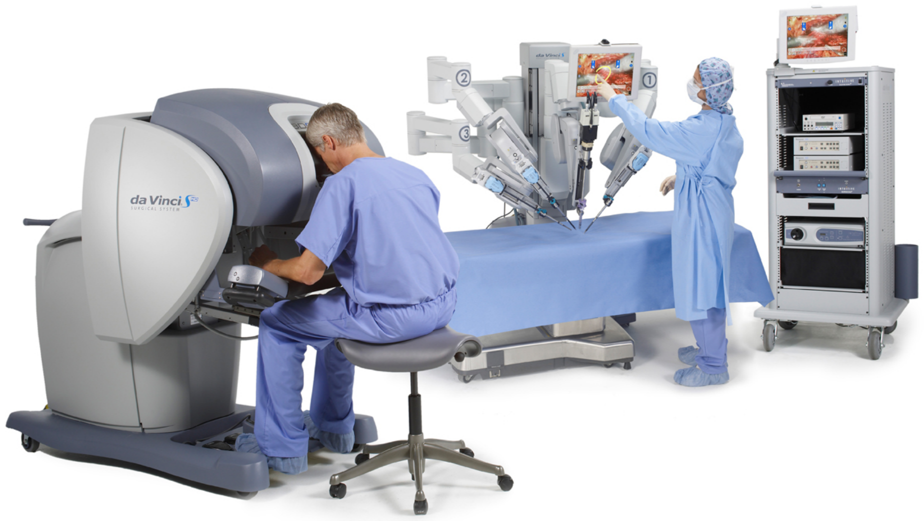


Figure 1.3: da Vinci Surgical System showing surgeons console, patient side cart and vision cart (©Intuitive Surgical, Inc.)

The console allows the surgeon to control the tools of the patient side robotic arms, while immersed in a virtual operating field. A stereo endsocope is used to provide a seperate video feed of the surgical site to the surgeon's eyes. This provides the surgeon with a 3D viewing screen,

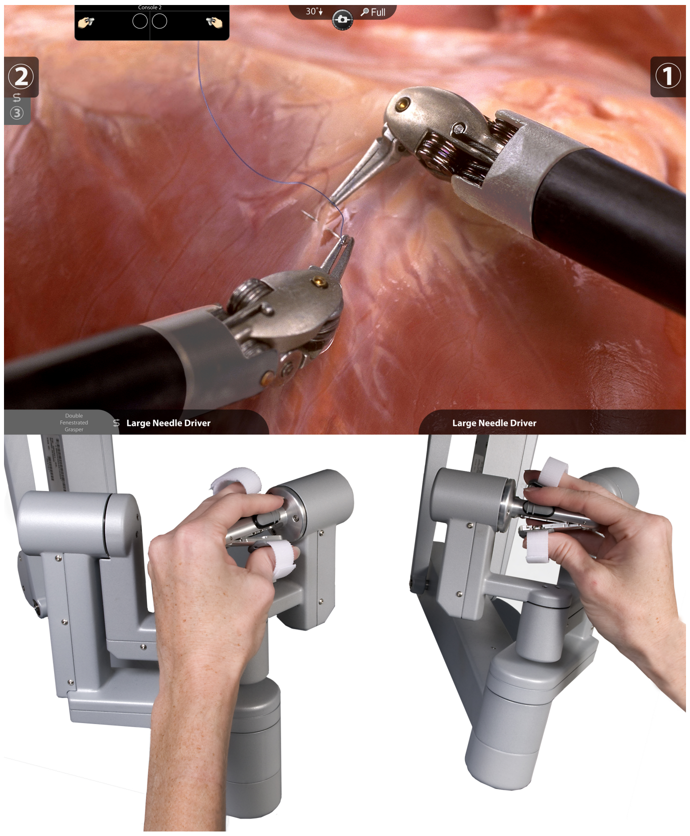


Figure 1.4: da Vinci master manipulator and viewing screen (©Intuitive Surgical, Inc.)

placed above the controllers to simulate the tools as an extension of the operator's hands. The stereo vision allows for depth perception, which has shown to improve operation time and decrease errors [16–18]. This console is used to control four robotic arms situated at the patient side cart for performing surgery through specialized EndoWrist®instruments. These instruments allow the surgeon to have natural dexterity while operating through the small incisions found in MIS, providing dexterity similar to the human wrist (Figure 1.6). Several foot pedals are also used to add functionality to the robot when operating, including clutching to increase the workspace size, and camera control.

1.1 Motivation

The da Vinci is teleoperated in a master-slave configuration, which takes the surgeons input at the surgeon console and translates it to the slave arms situated at the patient. The master controllers

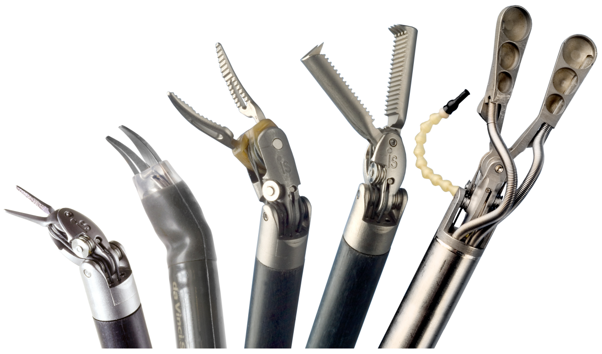


Figure 1.5: Different types of Endowrist® instruments; from left to right: needle driver, energy instrument, forceps, retractor, and cardiac stabilizer (©Intuitive Surgical, Inc.)

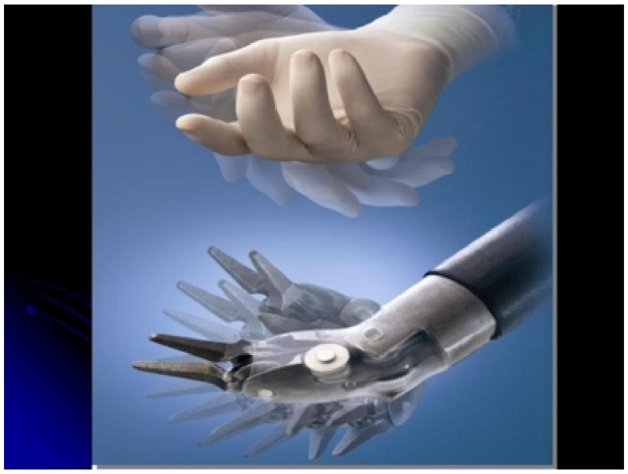


Figure 1.6: Comparing Endowrist® to the human wrist (©Intuitive Surgical, Inc.)

track the kinematic movements of the operators hands, and relay those movements to the robotic end effector (surgical tools). The point of interaction between the surgeon and the robot is called a human–machine interface (HMI), and is the method of relaying the hand movements to the robot. Since the HMI is the site for the flow of information between between the surgeon's hand movements and the robotic system, it has a direct impact on the overall performance of the system [19]. There has been recent research into novel HMIs to improve ergonomics and dexterity of the interface for RAMIS [20, 21]. A poll of surgeons found that the driving factors for an effective HMI were comfort and precision [22], while still allowing the range of motion required in surgery [23]. Many have suggested that the best interface is simple and free of technical detail [19]. Therefore, it is important to have an HMI design that is ergonomic for the surgeon, while not sacrificing performance. The current system has some limitations, including the large interface, not allowing the surgeon to operate as naturally as possible. The mechanical controllers can limit the wrist in certain orientations, and although designed to be intuitive, still requires extensive training to master.

The overall cost of the surgical system also plays a part in the progress of RAMIS. The da Vinci system ranges from \$1 to 2.5 million [11], and the current master console, or HMI of the da Vinci is very large, taking up operating room space. This large mechanical interface is not sterilizable, preventing the smooth transition between robotic and manual operation modes of surgery. The problems presented here have inspired on going research into different interaction methods in robotic surgery, with the attempt to make the interaction more natural and intuitive for the user. The solution of interest is computer vision (CV) based hand tracking, allowing the surgeon a noncontact interface with the robot. Current solutions have found that vision based methods have a suitable accuracy, but lack the robustness needed for surgery [21, 24]. Occlusion issues are the main issue found, causing inaccuracies in tracking performance. This has motivated the research presented in this thesis, with the design of a novel method of interaction, attempting to address the issues found presently in vision based tracking methods.

1.2 Project Goals and Contributions

The focus of this research is to investigate a novel HMI for use with the da Vinci surgical system using a noncontact approach utilizing infrared cameras. The research is done on the da Vinci, but can be applied to any RAMIS system. This approach will hopefully reduce some of the shortcomings with the current interface. Replacing the master console with a noncontact method, allowing the surgeon to operate with their natural hand motions, may pose an improvement over the current methods. The system created allows users to operate as if they were naturally picking things up with their fingers, while providing a sterile contact–free interface. Other vision based attempts have found that finger occlusion is an issue, so a part of this research was to address this problem, combining multiple view points of the hand. An analysis of the tracking workspace of the sensor is provided, and a solution that combines multiple sensors. Additional work is done on replacement of the large master console and viewing screen, with a head mounted display. Using a virtual reality (VR) headset, the stereoscopic viewing screen is placed directly on the surgeon, without the need for the large console. Finally, work is done on the control of the camera arm using the motion of the head.

The entire master console including the mechanical interface, and the large viewing screen, has been replaced with small infrared sensors, and a head mounted display. This greatly reduces the cost of the system, and the overall size in the operating room, while providing a sterile interface for the surgeon. The idea is that controlling the robotic tools with motions more natural to the user may result in faster learning for surgeons in training, and potentially increased performance.

1.3 Organization of Thesis

The structure of this thesis is summarized in the outline below:

Chapter 1 Introduction: The introductory chapter.

Chapter 2 Background: Overview of the current da Vinci interface, and other interfaces.

Chapter 3 Hand Tracking for RAMIS Teleoperation: The basics of the teleoperation of the da Vinci using the Leap Motion controller, as well as optimizing sensor placement.

Chapter 4 Endoscopic Camera Control: Control of the camera system, and providing a head mounted display of the surgical site.

Chapter 5 Validation: User study comparing the system with the da Vinci master console.

Chapter 6 Conclusions and Future Work: Highlights the contributions of this work. Recommendations for future work are also given.

Chapter 2

Background Information

The focus of this work is on the interface used to control teleoperated surgical robots. This chapter highlights the current interface used to control the da Vinci, and a literature review of the work done on human machine interfaces for teleoperation. Then an overview of the hardware and software used in the project is given. This was organized in this manner to provide the reader with the necessary information to understand the subsequent chapters. The work done can be applied to any RAMIS system, however the focus of this thesis is on the da Vinci robot, so a knowledge of the system is in order.

2.1 da Vinci Interface

Since the da Vinci is the main focus of the work, an overview of the surgeon's interface is given here. The master tool manipulator (MTM) of the da Vinci, is a mechanical linkage used to track the position of the hand, used to teleoperate the slave arms. The manipulator has 8 DOF (Figure 2.1), with the first seven joints actuated, where gravity compensation is used in order to reduce fatigue for the surgeon. The user grips the manipulator with the thumb and either index or middle finger, and is secured with adjustable loops. The movements of the hand are recorded by encoders in the first 7 joints of the MTM, and hall effect sensors are used for detecting the pinching distance. These signals are then translated into commands to move the slave robot in the same fashion. The operating range of each of the MTMs in the da Vinci is 35 cm $\times$ 45 cm $\times$ 35 cm. The interface for the surgeon also includes the 3D viewing screen, and the footpedals for control of the camera arm and various other functions.

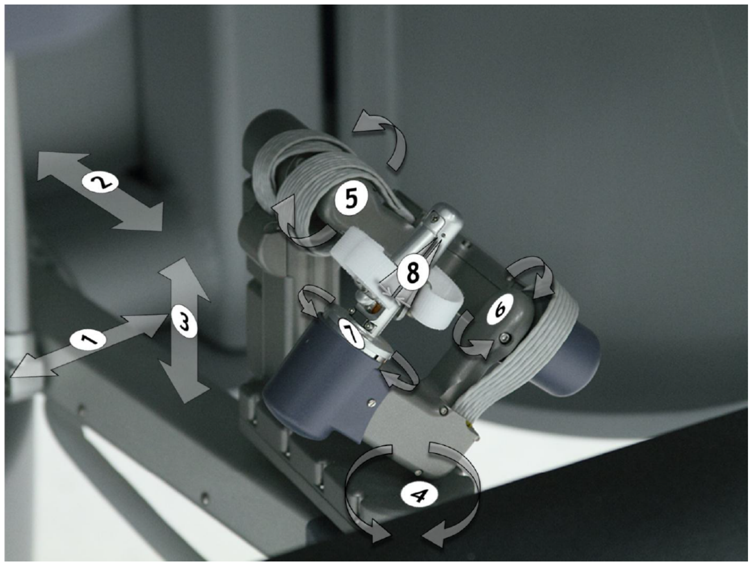


Figure 2.1: MTM joint axes of rotation (©Intuitive Surgical, Inc.)

While the current master console is fairly intuitive to use, the mechanical controllers can pose limitations on the natural movement and range of motion of the wrist. There may be other methods that provide a more natural interface for the surgeon. The large surgeon console takes up space in the operating room, and places the surgeon away from the patient. Various other methods have been researched for teleoperation, including glove based systems, and vision based tracking, without the need to contact a master device. The human machine interface (HMI) for robotic surgery is a key component in the performance of the surgeon. The next section highlights the literature in the area of HMIs and robotic teleoperation.

2.2 Literature Review

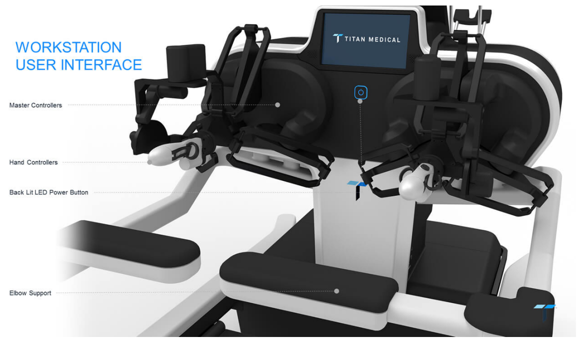
This section explores the current technology and research into human machine interfaces for robotic surgery. Also highlighted are other methods of tracking hand and finger motion that can be applied to surgical robotics. The techniques in the literature can be categorized as mechanical, glove based systems, and vision based tracking.

2.2.1 Mechanical Interfaces

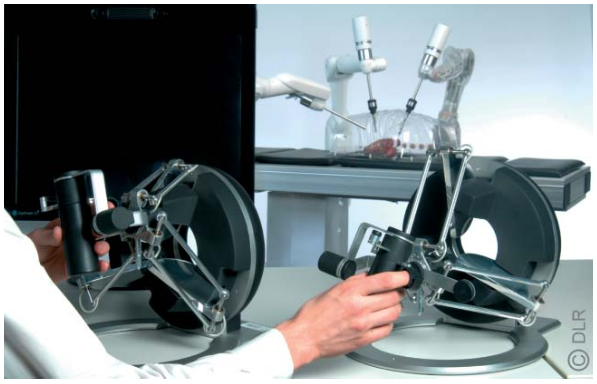
Mechanical interfaces that a surgeon manipulates to control the end effector are the traditional method of surgical teleoperation, including the da Vinci System. Mechanical linkages are used, and the kinematics of the linkage used to calculate the position and orientation of the end effector. These signals are used to command the slave robot to the deisred pose. Other surgical robotic systems have been designed with similar interfaces to the da Vinci. An example of a commercially available system is the SPORT Surgical System (Titan Medical, Inc., Canada) [25].(Figure 2.2a). This system is used in much the same way as the da Vinci, where the surgeon grasps the master interface between the thumb and index finger, and a 3D viewing screen is presented to the surgeon. The MiroSurge (DLR Institute of Robotics and Mechatronics, Germany) [26], developed in 2009, also uses a very similar interface to the da Vinci, using Omega 7 haptic device (ForceDimension, Inc., Switzerland)(Figure 2.2b). In 2013, De Donno *et al.* designed the STRAS (Single access and Transluminal Robotic Assistant for Surgeons) robot for endolumianal and transluminal surgery [27], and similarly used two Omega 7 master interfaces.

Greer *et al.* put a lot of research into the human machine interface (HMI) for robotic surgery, and developed the neuroArm, a full telerobotic surgical system. The interface for the surgeon can be seen in Figure 2.3. The hand controllers are designed to mimic holding a pencil, modelled after the Phantom Premium (SensAble Technologies, Inc.) (now Geomagic, Inc. USA). It includes a tool actuator mechanism the surgeon can control by pinching. The design focused on ergonomics, and to maximise the workspace for the surgeon. Other information was also available to the surgeon, including visual and auditory feedback. The group defined the HMI as anything that the user interacted with, and therefore the 3D screens, visual and auditory information all combine to create the HMI for the robot. The main focus was to provide an interface that was sufficient for performing the surgery without overwhelming the operator with information.

Another company has developed an interface that aims to mimic traditional laparoscopic surgery, where the surgeon is situated at the patient bed. The interface is much like a laparoscopic tool (Figure 2.4a). This would be beneficial for surgeons who have training in traditional laparoscopic surgery, however remains unintuitive for a novice user, and does not allow the same



(a) Sport Surgical System © Titan Medical Inc.



(b) MiroSurge Robot ©German Aerospace Center (DLR).

Figure 2.2: Examples of mechanical interfaces for HMI (a) Sport Surgical system; (b) DLR Mirosurge

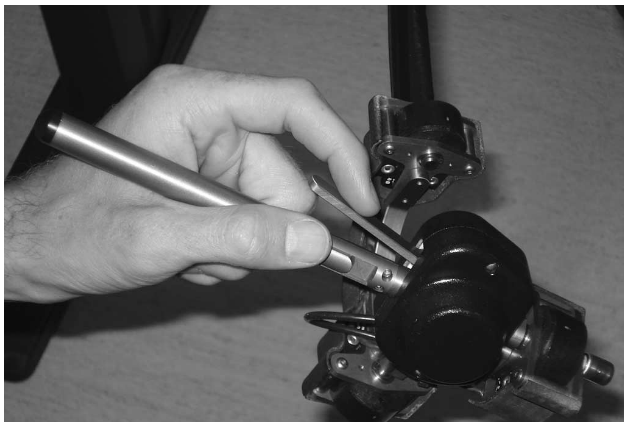
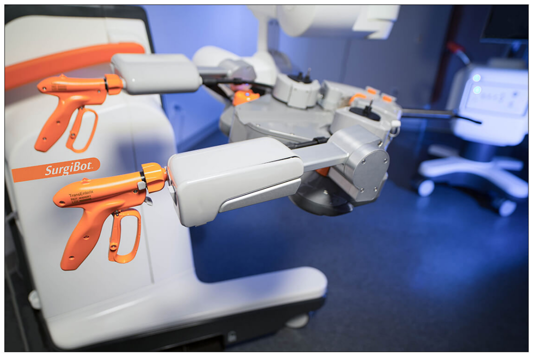


Figure 2.3: Neuroarm interface ©2008 IEEE [19].

range of motion and dexterity seen in the da Vinci. The same company has also developed a teleoperated version with a similar surgeon console like the da Vinci (Figure 2.4b). A similar interface was designed by Wortman *et al.* [28], where a miniature *in vivo* robot was designed capable of performing operation within the abdominal cavity. Originally controlled by the Phantom Omni controllers, Wortman *et al.* added onto this work, developing their own interface that mimics laparoscopic tools (Figure 2.5). However, the interface was found to restrict the surgeons movement when performing complex surgical tasks [29]. The Mantis Duo, developed in 2006, consists of two master grips that simulate the grips on the surgeon console for the da Vinci Surgical System. The masters are connected to a gimbal, allowing 7 DOF, including translation and rotation around the


(a) Surgibot System

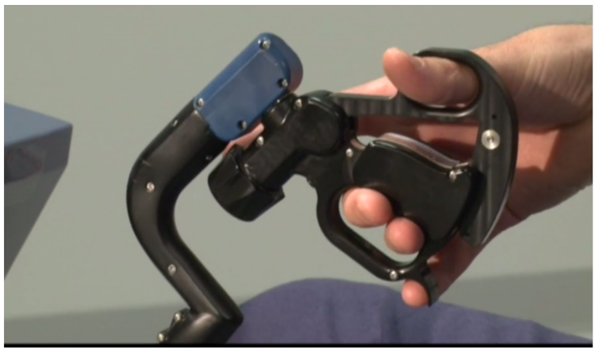

(b) Senhance Surgical Robotic System

Figure 2.4: Laparoscopic like interfaces, (a) Surgibot ©2017 TransEnterix; (b) Senhace Surgical System ©2017 TransEnterix.

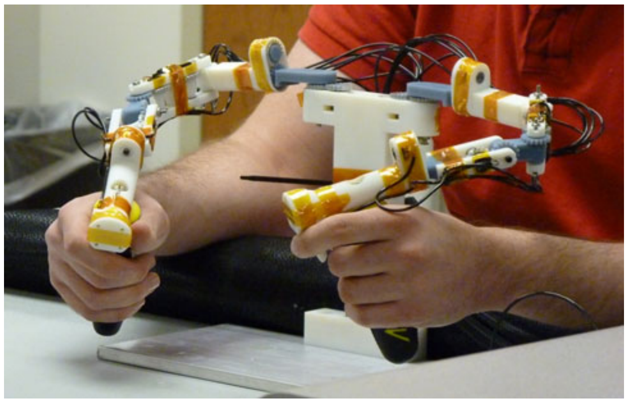


Figure 2.5: Wortman *et al.* interface ©2011, Springer Science+Business Media, LLC [29].

x, y, and z axes, as well as grip closure [30]. The gimbals are attached to the platform with cables under tension (Figure 2.6), used to measure the position of the grip. This interface was shown to be fairly intuitive for users, however, the wires can get in the way, limiting rotation in certain orientations [31]. This interface has been used to develop the Mimic®dV-Trainer®, used as a virtual reality surgical skills training and assessment device [32]. In 2009, Phee *et al.* developed the MASTER [33], which utilises a multi DOF joystick which maps the the movement of the user to the end effectors. This interface is also very similar to the da Vinci, grasping between the thumb and index finger, however, did not provide the same range of motion as the da Vinci controllers. The RAVEN, and then RAVEN II [34], were developed for use in telerobotics research, and Rosen *et al.* [35] used two commerical haptic controllers (Figure 2.7),(Geomagic, Inc. USA)) to control

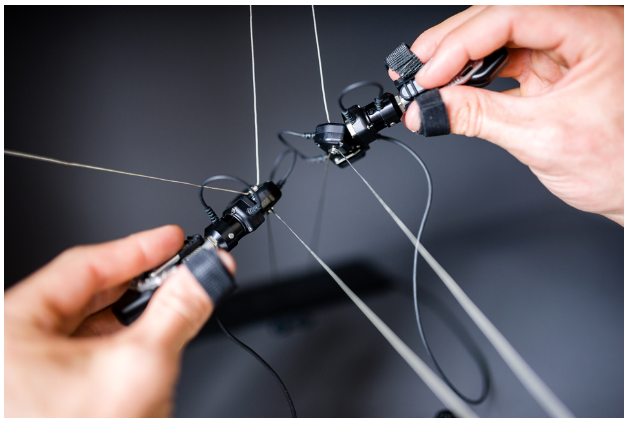


Figure 2.6: Mantis Duo interface showing cable connection (©2016 Mimic Technologies)

the end effectors. Two buttons on the pen were used to control the open and close position of the gripper. Visual feedback was the only method of controlling the amount the gripper was open and closed, and was not very intuitive for the user. Several other commerical haptic master devices have been used to control the RAVEN robot [36].

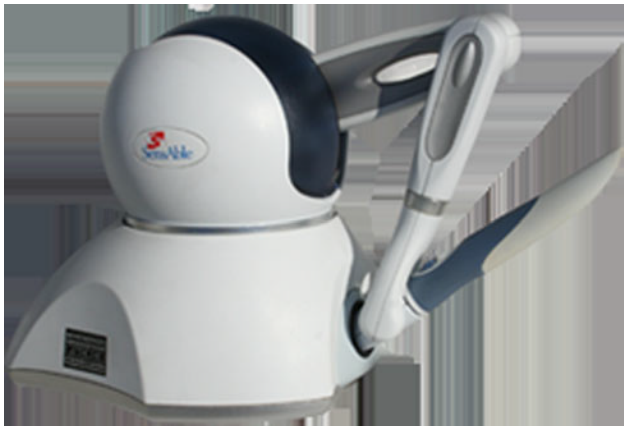


Figure 2.7: Phantom Omni® haptic device (©Geomagic, Inc.)

All of these are examples of interfaces where mechanical linkages are used with an interface that the surgeon grips. The position and orientation of the end effector are computed using the kinematics of the linkage, and used to control the slave position and orientation. They all have the same issues, in that they can mechanically restrain the movement of the hand, and are not the most intuitive to use. The pose of the hand is constrained to match the tool used, and may not be the most natural orientation for the user. Taking the natural motions of the hand as input for human computer interaction (HCI) has been a part of ongoing research for the past few decades. These methods may pose an improvement over the current mechanical interfaces.

2.2.2 Gesture Tracking

There has been significant work done using hand gestures as a control in HCI. The hands are used as a means on nonverbal communication, and therefore can be used as an intuitive method of interaction with computers. The methods for tracking hand motions can be divided into two broad categories: vision based, and contact based. Contact based methods involve physical interaction with the user, while vision based methods analyse images from one or multiple cameras for inferring hand and finger position.

Contact based includes data gloves that track the motion of the hand and fingers with various sensors. There have been many data gloves developed for tracking hand motion over the past 30 years [37]. Some of the first prototype gloves include the Sayre Glove, and the LED glove developed at MIT [38]. The Sayre Glove used flexible tubes with a light source at one end and a photocell at the other. The amount of light passing through decreased as the finger was bent, and therefore the voltage of the photocell corresponded to the amount the finger was bent [38]. The MIT-LED glove used LEDs studded on a cloth glove for tracking hand motion for computer animation. These two gloves were hard–wired, and very obtrusive and cumbersome, and did not get used for very long, but were important for the development of research in the field. The Digital Data Entry Glove, designed at Bell Telephone Laboratories in 1981, was designed for data entry using the Single–Hand Manual Alphabet [39]. It was programmed to recreate alphanumeric characters from 80 different hand positions. It used proximity sensors to sense when fingers were pinched together, "knuckle bend sensors" for measuring the flexion of fingers, and tilt sensors for the tilt of the hand in the horizontal plane [40]. In 1987, Zimmerman developed the Data Glove, designed for general purpose applications that use hand and finger motions as input [41]. This glove was a large improvement over the other designs, as it operates in real–time, and was much lighter, and less obtrusive. Like the Digital Entry Data Glove, it also used flexible tubes, with a light source to record joint angles. This glove was later commercialized, with a optical fibre system. This glove's popularity boosted research in the area of teleoperation and robotics. The first project using this glove for teleoperation was by AT&T, where the glove was used to control a dextrous hand. Hong *et al.* used the Data glove for teleoperation of the Utah/MIT hand [42]. The motions of the hand were mappped to control signals to control the dexterous hand designed by Jacobsen *et al.* [43]. While the glove was accurate enough for simple tasks, the finger flexion accuracy was only accurate to approximately 10 degrees, not suitable for more precise tasks [38]. Pao *et al.* [44] similarly used the DataGlove to control the same dexterous hand, and found while able to track the basic pose of the hand, there were limitations, due to the limited number of sensors, and sensor resolution in the glove. They concluded that there would need to be improvements in the glove in order to accurately track the hand for any pose.

The CyberGlove (Figure 2.8), developed in 1992, uses piezo–resistive sensors to measure the bending of the fingers. It comes with 2 bend sensors on each finger to measure extension and

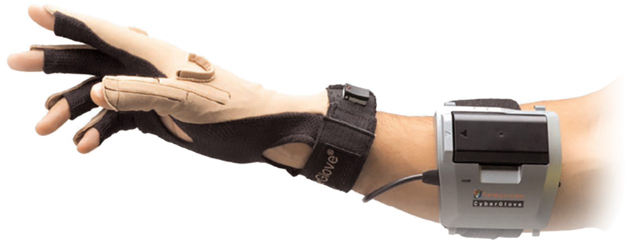


Figure 2.8: Cyberglove (©CyberGlove Systems)

flexion, as well as additional sensors to measure adduction and abduction. The CyberGlove has been used in teleoperation by Iba *et al.* [45], who used the device for controlling a mobile robot. Communication was done between the human and robot using 6 gestures: opening, closing, pointing and waiving left or right. Theses gestures corresponded to commands such as stop, or turn left or right. Griffin *et al.* used the CyberGlove in analysing the kinematics of the hand to control a planar manipulator and found difficulties with calibrating the device for different users. More recently the glove was implemented with the NASA/Defense Advanced Research Projects Agency (DARPA) Robonaut, an anthropomorphic human scale robot [46]. One Cyberglove on each hand was used to control the robot's arms and dexterous hands, as well as the camera system. The user could teleoperate the robot to perform tasks such as tightening a bolt and tying knots.

There have been numerous other glove systems, using various sensing techniques. The Humanglove, commercialized in 1997, uses hall effect sensors to measure the flexion and extension of each finger. The 5DT Data Glove [47] uses proprietary optical fibers to measure the overall flexion of the fingers, however is not suited for complex hand poses. Various other data gloves can be found in [37]. Hernandez-Rebollar *et al.* [48] developed the AcceleGlove, which uses six dual–axis accelerometers mounted on the fingers and the back of the palm. They report the position relative to the gravity vector. It was designed for manipulating objects in virtual reality, and for interpereting sign language. Baldi *et al.* [49] used 6 interial measurement units (IMU) placed on the fingers and hand to track the position and orientation of the hand and two fingers. The data from accelerometers, gyroscopes and magnetometers were fused to provide the orientation of the hand, however sensitivity to magentic fields was an issue. The benefit of these gloves is that the orientation of the hand is known, and not just the shape of the bending of the fingers. However, this comes with the trade off of tracking precision [49]. Cortese *et al.* used the Acceleglove to teleoperate a mechatronics gripper for hand rehabilitation [50].

Although these glove systems all differ in terms of the sensor technology and placement, they all have the same basic idea, where sensors are integrated into a cloth glove, and bending of the finger joints is measured. They also all have the same issues, being that they are tethered, which can restrict motion, and require calibration for different users. Many of the gloves are large and uncomfortable, and the cloth support has been found to affect the measurement performance [51], limiting the precision in pose estimation. Recently, work has been done by Tran *et al.* [52] to create a wireless data glove for hand gesture based robotic control, attempting to reduce the restriction of being tethered. Similar work can be found by Kim *et al.* [53], and Kumar *et al.* [54]. Although an improvement in portability, the added sensors and circuitry placed on the hand and wrist make the gloves heavier, and more cumbersome to use.

Another form of glove is an exoskeleton type of glove, where mechanical linkages are used to measure the flexion of the fingers. An example of this is the Dexterous Hand Master (DHM) [55]. The DHM is an exoskeleton–like device worn on the hand, where finger flexion is measured with sensorized mechanical linkages on each of the fingers (Figure 2.9). Hall effect sensors are used as potentiometers to measure the flexion angle of the three joints of each finger [39]. The work done at AT&T with the Data Glove was later moved to using the DHM, due to the kinematic similirities

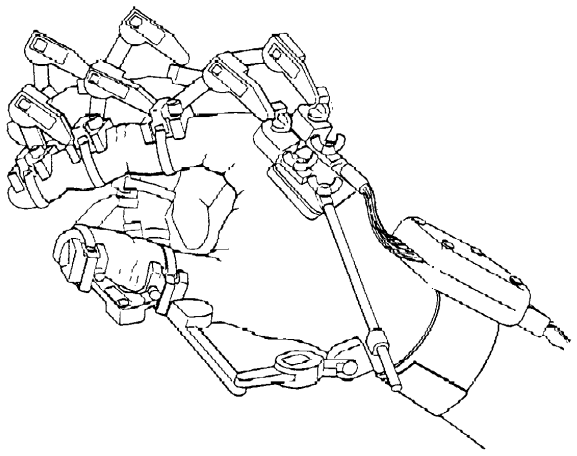


Figure 2.9: Dexterous Hand Master

between the master and slave devices [56]. Marcus *et al.* also used the DHM to control the dexterous Utah/MIT hand [57]. The DHM ws very accurate, with a 92 to 98 percent correlation between the real finger positon and the DHM sensor position, however, the device required a tedious and slow calibration for different users, and instability arised during rapid motions [58,59]. The device was also very bulky. Bouzit *et al.* [60] designed the Rutgers Master II, which is another wearable exoskeleton, which uses pneumatics to provide haptic feedback, and the joint positions are measured with IR and Hall effect sensors.

Recently, Olsson *et al.* designed a haptic glove for interaction with a virtual reality simulator [61]. However, it only provided a single degree of freedom, and was integrated with the commercially available Phantom Premium haptic controller. In combination, this provided the 7 DOF needed for teleoperation. Having the hand attached to the mechanical links of the Phantom device, movements were restricted in the roll and pitch directions.

The main issue with all of the glove type systems reviewed so far, with the exception of the Acceleglove, is they only measure the relative motion of the fingers. To obtain the necessary 7 DOF for teleoperation with the surgical robot, external tracking systems must be used to obtain the global position and orientation of the hand. The CyberGlove provides an attachable mount for a mechanical arm which can be used for force feedback applications (Figure 2.10), providing the orientation and position of the hand relative to the base, as well as a magnetic tracking system that can be mounted on the wrist. The issue with magnetic tracking is that it is sensitive to magnetic

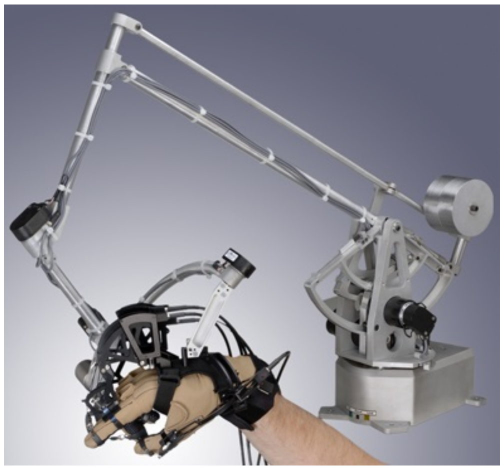


Figure 2.10: External positioning for cyberglove (©CyberGlove Systems)

fields, and interference in the nearby area (operating room) can cause issues with performance. Ferromagnetic materials need to be removed from the area near the transmitter and receiver. The Data Glove mentioned previously also used magnetic tracking for position of the wrist, and a less expensive version of the glove used an ultrasonic tracking system, where two ultrasonic transducers were placed on either side of the metacarpals to measure the position, roll and yaw of the hand. Testing showed that these glove systems were only accurate to approximately 10 degrees of rotation [62].

2.2.3 Vision Based Tracking

The glove based systems can provide the necessary DOFs for teloperation, however there are drawbacks in that they hinder the naturalness in which the user can interact with the robot, and can be obtrusive and expensive. For this reason, considerable research has been done into tracking the hand in a non–contact method using vision based tracking or computer vision (CV). Vision based tracking of the hands has been in development for about two decades [63, 64]. Images of the hand are used to extract the hand pose and position using one or more cameras. Many of the first attempts used CV to estimate a rough pose of the hand, used for gesture recognition. The hand pose is compared to predefined gestures such as pinching, or pointing [65]. Several reviews of the work done in the field of hand gesture recognition can be found in [66], [67], [63]. A single camera was found to be effective in recognizing gestures [68], [69], however in order to obtain depth information, multiple cameras are needed. These works mainly focused on recognizing pre defined gestures, however, to perform tasks in real time such as teleoperation, the real 3D kinematic motion of the fingers and hand is required. The movement of the hand needs to be tracked in 3D space, to map movements to a robot. Erol *et al.* [65] provides a review of the technology and techniques used in hand kinematic modelling and tracking up to 2007. Some of the main issues with these systems were the sampling rate and processing speed. Sturman *et al.* [70] recommends that at least 100 Hz tracking speed for effective interaction or teleoperation, whereas the fastest system reviewed in the literature was 30 Hz [71].

With the advancement of commercial optical sensors such as the Kinect (Microsoft, In., USA), there has been numerous research projects in hand tracking and HCI. Various work has been done in developing robust real–time hand gesture recognition algorithms for the Kinect sensor [72], [73], and these have been used to control a robot with hand gestures [74]. Gesture recognition has also been used for a gesture driven scrub nurse, with the task of delivering surgical tools in the operating room [75]. Others have used hand tracking for interacting with medical software in a sterile manner [76–78]. Kim *et al.* [79] compared the performance of the da Vinci robot when teleoperation was performed with the Kinect, and other touch based interfaces. They concluded that the high latency, and accuracy of the sensor caused the system to underperform compared to the traditional interface. Another relatively new sensor, the Leap Motion controller (Leap Motion, Inc., USA), is a device that uses infrared (IR) light to natively track all 10 finger and hand position with submillimetre accuracy, with a refresh rate up to 120 Hz. [80]. Bassily *et al.* [81] used the Leap Motion controller to teleoperate a 6 DOF robotic arm for use as an assistive device. The idea was to incorporate robotics into homes to help people with disabilites. Steretu *et al.* [82] used the Leap to control an anthropomorphic hand, finding that there was a natural interaction between the user and the robot with a high degree of accuracy. Zubrycki *et al.* [83] compared both the Leap Motion controller and the Kinect to control a 3 finger gripper. Despinoy *et al.* [21] compared the performance of the Leap Motion controller and the Sigma 7 haptic device (Force Dimension, Switzerland) as an HMI for teleoperation of the Raven-II robot. Preliminary results show that the IR sensor is comparable in performance to the mechanical device, although only simple tasks were performed. More complex tasks may result in greater self occlusions of the fingers.

Most recently, Zhou *et al.* [24] evaluated different modes of controlling the Taurus robot, a small dexterous robot with two 7 DOF arms capable of high precision and micro movements. The interfaces included the Leap Motion controller, Kinect sensor, and two contact haptic controllers. Two experiments were completed to measure the performance on the various interfaces. The first was to make an incision following a curved path, and the second a peg transfer task. The Leap was shown to have similar performance compared to the contact controllers in the incision task. And in the peg transfer task, while initially the completion time was longer, it showed to have a much faster learning rate compared to the others interfaces. The number of times the object was dropped was measured as well, and the two non–contact interfaces did not perform as well. The authors attribute these "mistakes" to losing tracking of the hand, causing unwanted opening of the tool. In order to improve this, occlusion issues need to be addressed with the vision based control methods. However, the performance of the two touch-less methods were shown to be more desirable in certain criteria, showing the potential for their use as a non–contact HMI in surgery.

2.2.4 Summary and Conclusions

There have been many different methods for interaction with robotics. Mechanical systems force the user to adapt their hand orientation to manipulate the device, and may pose limitations on the orientation, but are very accurate. Glove based systems also make contact with the users hand. They can be awkward to use, and do not provide the same accuracy as the mechanical devices. External tracking systems are also needed for the global position of the hand, and have their own limitations. More recently, vision based methods have been focused on, and promising results have been shown in several cases. An interface that allows the surgeon to manipulate the tools in a manner that is natural without making contact with a device is the benefit of this approach. However, occlusion issues, and the robustness of the tracking algorithms need to be improved upon. The research here has motivated the research of this thesis, in creating a novel contact–free interface free of occlusion issues.

2.3 Hardware and Software

This section highlights the hardware and software used in this research. The da Vinci Surgical System is a proprietary product, and therefore provides limited access for research. In order to promote the advancement of telerobotics research, a research platform was developed for the da Vinci. The da Vinci Research Kit (dVRK) [84], combines open source software and electronics, which allows research institutes access to all levels of control of the robotic system. The dVRK is built on the open source cisst/SAW libraries from Johns Hopkins University [85]. A Robot Operating System (ROS) interface is also available, which allows for easier communication with the various components of the da Vinci. The dVRK is built for the original da Vinci system, and interfaces with the commercial hardware with custom controller boards. The system is a fully functional telerobotic system, allowing different methods of hardware and software integration. The dVRK controllers available for this project are connected to a PC running Linux (Ubuntu), with a 3.4 GHz quad core i7 processor and 16 GB RAM.

2.3.1 da Vinci Hardware

The dVRK platform interfaces with the hardware from the original da Vinci system, consisting of the surgeon console and the patient side manipulator (PSM). The surgeon console includes a binocular viewfinder and stereoscopic viewing screen, two MTMs, and a footpedal tray (Figure 2.11). The patient side cart (Figure 2.12), houses three 7 DOF patient side manipulators with surgical tools, as well as the endoscopic control manipulator (ECM) for holding the endoscopic camera. Only two of the PSMs are used in this research.

The PSMs have 7 actuated joints, and joint sensors for control purposes (Figure 2.13). The first three joints control the position of the tool (outer yaw, outer pitch, and insertion of the tool), and Joints 4–7 control the orientation and opening and closing of the tool's jaw. The latter 4 joints interface with the Endowrist instruments, actuating cables that control the joints of the tool tip. The position controlled by the first three joints can be seen in Figure 2.14, where the wrist position is the control point. The next four joints are then used to control the orientation of the tool, defining the position of O_7. Joints 6 and 7 work in combination to control the rotation angle

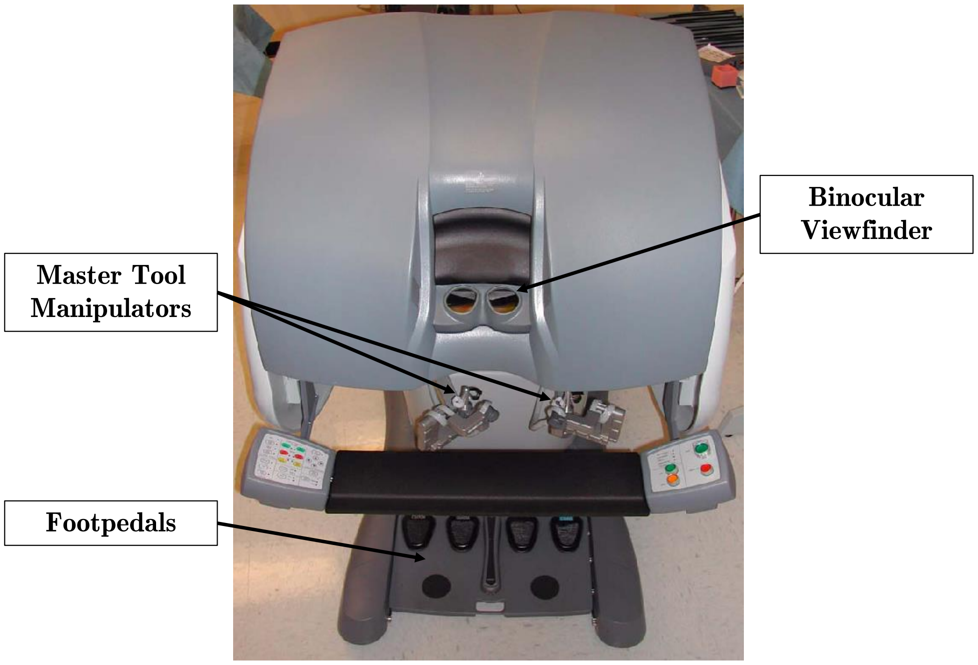


Figure 2.11: da Vinci surgeon console

at O_6, as well as the gripper opening angle.

There are also 5 passive set up joints for positioning the robot arm for surgery. These joints are used prior to surgery, and do not move during the procedure. The ECM also includes 3 passive set up joints, and 4 actuated joints providing 4 DOF. The ECM holds a stereo endoscope with a pair of cameras, used to generate a different image for each eye. The separation between the cameras produces the immersive 3D stereoscopic view for the surgeon.

2.3.2 ROS Interface

ROS (Robot Operating System) [86] is a pseudo-operating system that provides a set of libraries and tools designed to help software developers create robotic applications. It allows for communication between robotic processes on a single computer or across a network of computers. The fundamental concepts of ROS are nodes, topics, and messages. ROS is designed such that is modular, in that different processes of a system are broken into a separate modules, or *nodes*. Nodes are the major processes of a system that perform computation, and communication occurs between different nodes by passing messages. A *message* is a simple data structure, including primitive

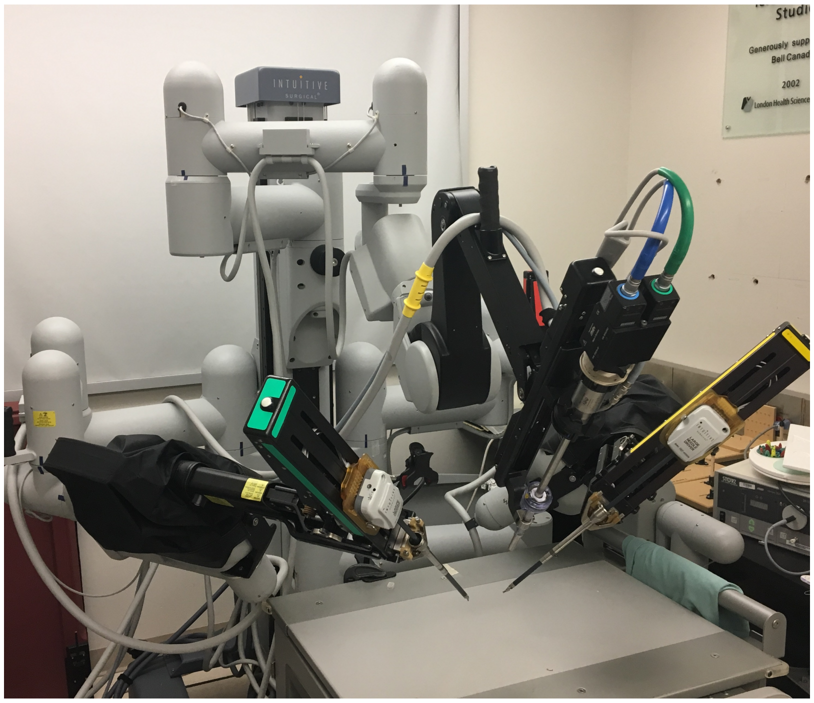


Figure 2.12: da Vinci patient side cart

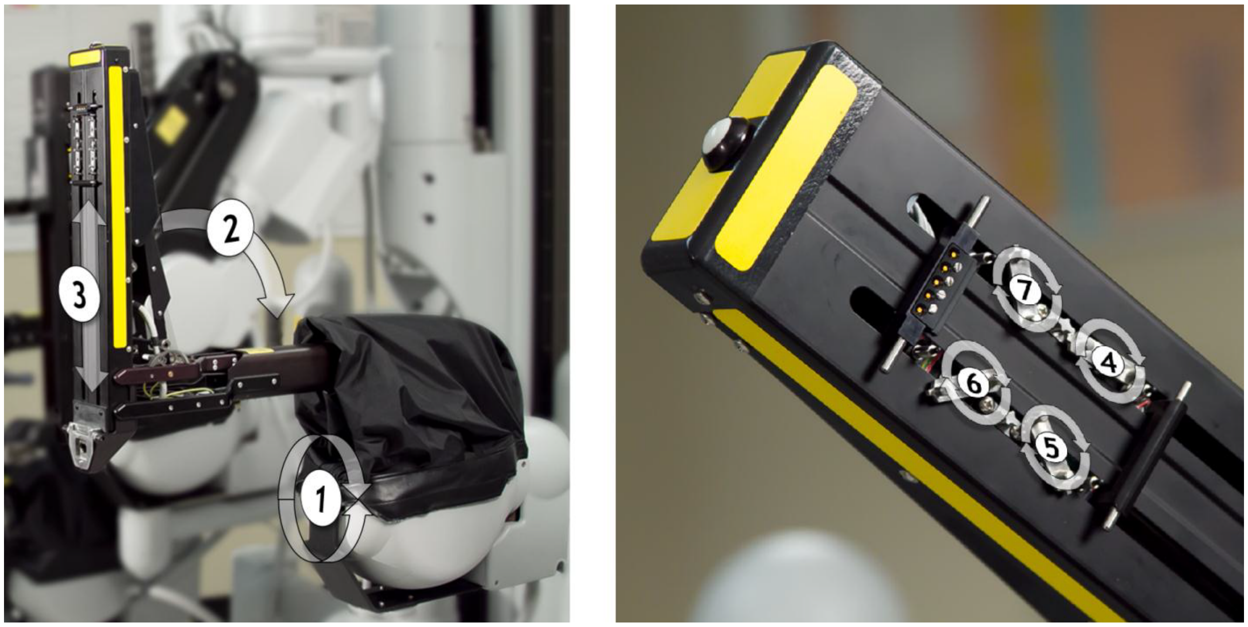


Figure 2.13: Joints of the PSM and their direction of motion (©Intuitive Surgical, Inc.)

types (integer, floating point, boolean etc.), as well as nested structures similar to a struct in C. *Topics* are the buses in which messages are sent between nodes. They are simply a string such as

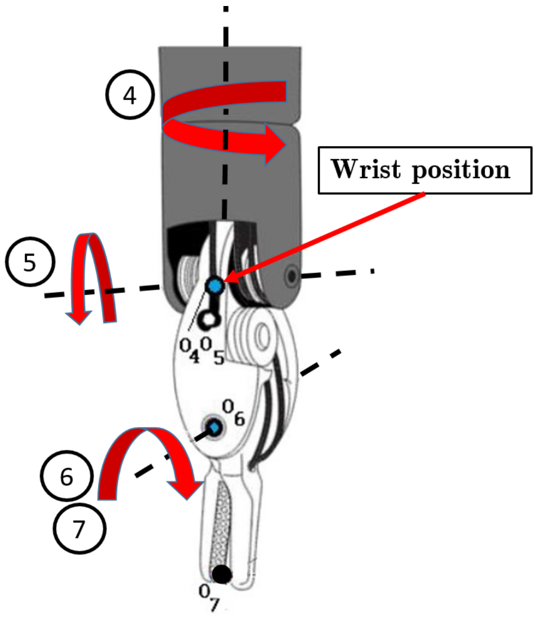


Figure 2.14: Endowrist joints and axes of rotation

'teleop' or 'sensor_data'. To send a message, a node publishes a message of the specific data type to the desired topic, and a node that is interested in specific data will subscribe to the appropriate topic. The relationship is anonymous, meaning the subscribing node does not need to know where the data are published from, and any data published to the topic will be subscribed. Messages are handled in the node by functions known as callbacks. These callbacks are automatically called when a message becomes available. If a node is subscribing to a topic, as soon as a message is published on that topic, the callback function is passed the message, and executes the function, performing the desired computation on the message. The callback can then publish a message on an additional topic. In this project, ROS is used for communication between the various hardware.

2.3.2.1 ROS–dVRK

The dVRK has been developed with a ROS interface, so components publish the robot state in several ROS topics, such as the position and orientation of the PSM, the gripping angle, the mode of operation etc. The interface is designed to also subscribe to ROS messages, enabling a developer to send commands through ROS, allowing users to control the manipulators. A list of the available topics of interest can be found in Table 2.1. The commands for controlling the pose of the manipulator are translated into the desired joint angles through the inverse kinematics of the manipulator, provided by Intuitive Surgical, Inc. The joint positioning is then done through PID control. A graphical user interface is provided with the dVRK software for interaction with the robot, providing buttons such as homing the robot, and adjusting of the PID gains.

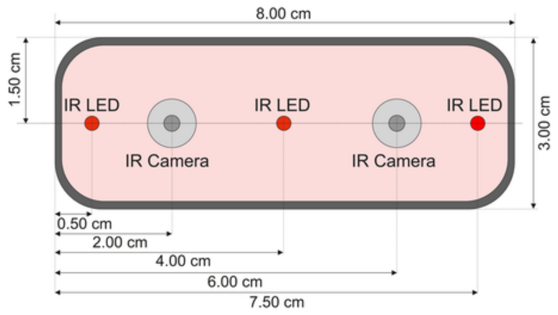
Table 2.1: List of dVRK Topics

Topic	**Type**	**Description**
/PSMx/set_robot_state	Subscriber	control mode of PSM
/PSMx/position_joint_current	Publisher	current PSM joint angles
/PSMx/position_joint_desired	Publisher	last joint angles sent
/PSMx/position_cartesian_current	Publisher	current pose of tool wrist
/PSMx/position_cartesian_desired	Publisher	last pose command sent
/PSMx/set_position_cartesian	Subscriber	move PSM to desired pose
/PSMx/set_position_cartesian_goal	Subscriber	move PSM to pose with smooth trajectory
/PSMx/goal_reached	Publisher	goal reached event
/PSMx/set_jaw_position	Subscriber	gripper jaw angle
/footpedals/coag	Publisher	safety pedal event
/footpedals/camera	Publisher	clutch pedal event

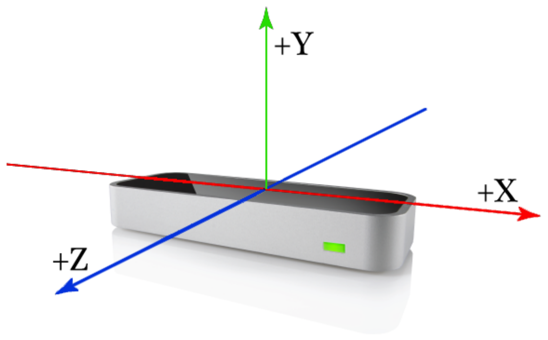
2.3.3 Leap Motion Controller

The Leap Motion controller was chosen as the sensor for hand tracking. It was chosen for its high refresh rate of approximately 100 Hz, large interaction workspace, and submillimetre accuracy [87]. Latency is also important in surgical teleoperation, and the leap motion controller excells over other similar devices, with an average latency of 35 ms in 2014 [88], which has since improved with USB 3.0 support. The device natively tracks finger and hand motion, unlike other comparable depth sensors, where a custom algorithm would need to be developed. The Leap Motion controller is a consumer grade desktop sensor used for tracking hand gestures and finger motion. It is primarily used for hand gesture interaction with software on a personal computer, as an alternative to using a standard computer mouse. The advantage being that there is no contact between the hands and the sensor. It uses a set of two monochromatic IR cameras, and three IR LEDs to provide information on the position and movement of various points of the hand. It is capable of simultaneously tracking the position of one or two hands, and up to 10 fingers. The positions are reported in mm in a right handed Cartesian coordinate system centred on the top face of the device. The x and z axes are in the horizontal plane, with the x axis parallel to the long edge of the device, and the z axis towards the user. The y axis is vertical with values increasing upwards. Figure 2.15a shows a schematic view of the device, and Figure 2.15b shows the coordinate system used. According to the manufacturer, the sensor's accuracy for fingertip position detection is 0.01 mm [87]. The cameras used incorporate wide angle lenses to generate a large workspace or interaction area, which can be visualized as an inverted pyramid. The optimal tracking area is between 25 mm and 600 mm above the controller [87]. Figure 2.16 shows the interaction area of the Leap Motion controller. The images captured by the cameras are sent to the host PC over USB, and all the computation is done by the dedicated software installed on the host machine. Since no computation is done on the Leap Motion controller, this allows for the high refresh rate of the sensor. Although Leap Motion does not disclose the working principle behind the device, based on the hardware, the sensor can be categorized into an optical tracking system based on stereo vision principles. Images from each of the cameras are slightly offset, and can be used to compute the distance of trackable objects from the sensor.

Several studies have been done assessing the performace of the leap motion controller [80, 89], finding that the accuracy of measured positions of a static object was less then 0.2 mm, and an average accuracy of 0.7 mm for gesture based interactions. This performance was much better than other comparable sensors such as the Kinect sensor. The Kinect is used primarily for full body tracking, whereas the Leap Motion controller is designed specifically for hand tracking.



(a) Schematic view



(b) Coordinate system (©Leap Motion, Inc.)

Figure 2.15: Leap Motion controller

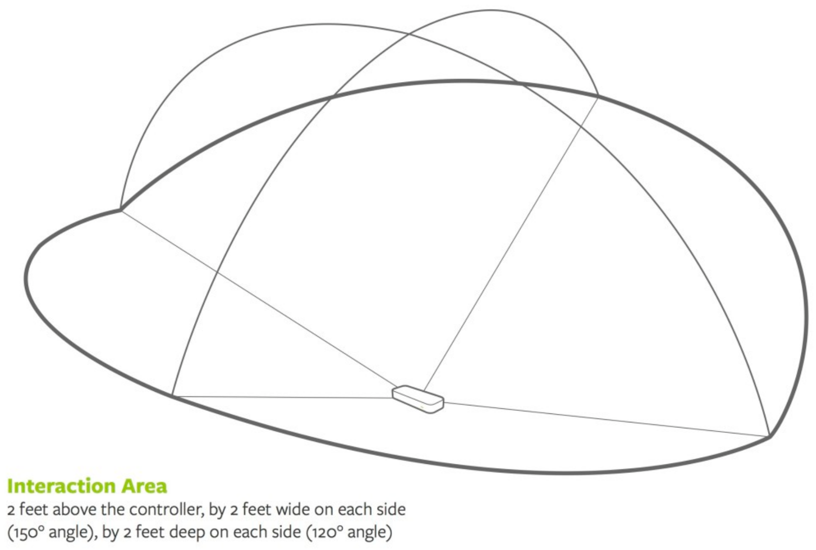

Interaction Area
2 feet above the controller, by 2 feet wide on each side (150° angle), by 2 feet deep on each side (120° angle)

Figure 2.16: Leap Motion controller interaction area (©Leap Motion, Inc.)

The Leap Motion Controller has an API that is available in many programming languages including C++ and Python. The C++ API was used in this research in order to interface with ROS. As the device tracks the hands in view, the tracking data are returned in the form of a frame object which includes all the information about the tracked objects at a single moment in time. These frame objects are updated at the frame rate of the device, ≃100 frames per second (fps), and are the most basic level of the tracking data. Within the frame objects there is a nested hierarchy providing more detailed information of tracked objects. All the tracking data begin with the Hand class. A Hand object provides information about the hands in view, including position, orientation, velocity, palm and wrist position, and a list of fingers associated with the hand. There are also a variety of public member functions, including the hand ID, timestamp, and hand confidence. The confidence rating is how well the observed data match an internal model of a hand, and can be used as an indicator of when the device is not tracking well. The finger class provides the position and direction of each finger, identified by type name thumb, index, middle, ring and pinky. The fingers are tracked with what Leap Motion calls skeletal tracking. Each finger includes the class Bone, where each bone of the finger is tracked. Each finger is modelled as a series of 4 links: including the metacarpals, the proximal phalanges, intermediate phalanges, and distal phalanges. Each bone object has public functions associated with it, including the bone length, direction, and the position of the start and end of the bone. These can be used to get the position of finger joints of interest. The joints of the finger starting from the tip are the distal interphalangeal joint (DIP), the proximal interphalangeal joint (PIP), and the metacarpophalangeal joint (MCP). Figure 2.17 shows the tracked points of the hand, and Figure 2.18 shows the hierarchy of the hand data tracked by the Leap Motion controller.

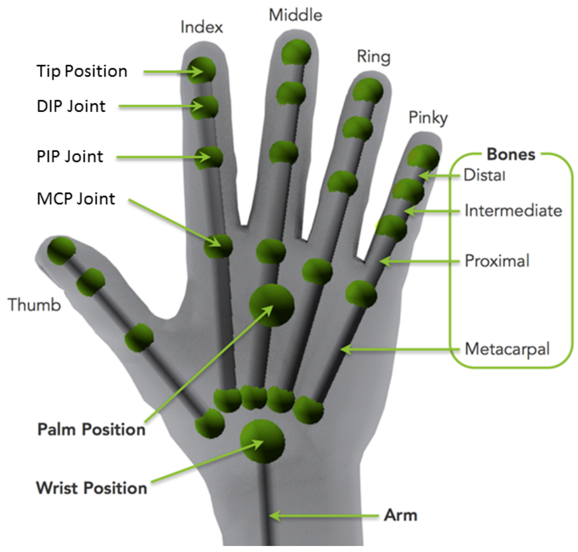


Figure 2.17: Leap motion tracked points (©Leap Motion, Inc.)

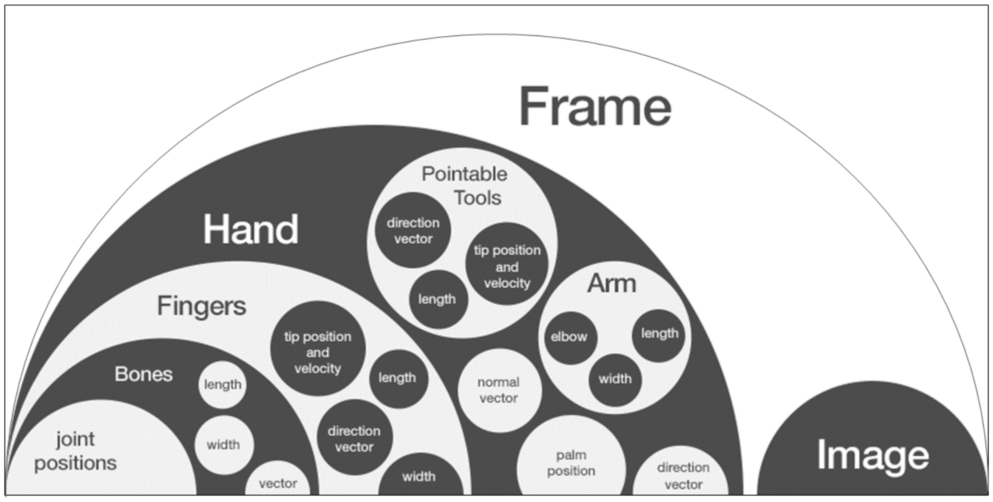


Figure 2.18: Leap Motion tracked data hierarchy (©Leap Motion, Inc.)

2.3.3.1 LeapROS

In order for communication between the Leap Motion controller and the dVRK, a ROS node was created for interfacing with the sensor. This is written in C++, and polls the sensor at a rate of 100Hz. In traditional operation of the Leap Motion controller, the frame rate is not fixed, depending on the resources available on the PC it is connected to. For teleoperation with the da Vinci, it was desirable to have a constant frame rate for control purposes and synchronization of data. This method of polling the sensor ensures that the frame rate is constant. The data from the hand are published with a custom ROS messages `leap_msgs:Hands`, on a separate topic for the left and right hands. The message holds the position of the joints of the hand, the palm and wrist position, as well as the orientation of the hand and a timestamp. Other nodes are able to subscribe to this topic, receiving the data from the controller at 100 Hz.

2.4 Conclusions

The human machine interface for surgical robotics is not perfect, and attempts have been made utilizing different methods of interaction with robotic systems. Prior work highlighted the potential of vision based tracking, and led to the motivation of this work, providing a noncontact interface for surgery. The components of the system used in this work have been outlined, allowing the understanding of the following chapters.

Chapter 3

Hand Tracking for RAMIS Teleoperation

3.1 Introduction

The following chapter highlights the implementation of the Leap Motion controller to control the surgical tools of the da Vinci using vision based hand tracking. The methods used to control the tool's pose with the data from the hand are presented. An analysis of the workspace of the leap controllers is provided, giving insight to where occlusion problems occur, and a solution by integrating multiple sensors. The final sensor placement for dealing with occlusion issues is determined, and the resulting improvement in tracking shown.

3.2 Teleoperation

Teleoperation with the da Vinci involves the control of 7 DOF of the patient side manipulator (PSM). The first 6 DOF include the position and orientation of the tool, with the 7^{th} DOF being the gripping angle of the jaw. The position and orientation of the PSMs can be controlled through the ROS interface on the topic "`/dvrk/PSM/position_cartesian_current`". This topic takes a built in ROS "Pose" message, composed of a quaternion, and a 3D Cartesian coordinate, representing the orientation and position of the tool with respect to the home position of the

PSM. Further mention of pose will be referring to the combination of orientation and position. Quaternions are a set of numbers that extend the complex number system, and describe the orientation of an object in 3D. In this thesis, rotation matrices are used in place of quaternions for ease of understanding. Pose commands are used to compute the desired joint angles of the PSM to place the tool at the desired position and orientation through the kinematic equations for the manipulator. The gripping angle can be controlled by publishing the desired angle on a separate topic, `/dvrk/PSM/gripper`. A combination of the position, orientation, and gripping angle commands will control the 7 DOF needed for teleoperation, and are provided by the data from the hand to control the robotic tools at the refresh rate of the Leap Motion sensor.

3.2.1 Orientation

The orientation of the end effector of the PSM can be defined by the rotation matrix $\boldsymbol{R}_e^b$, which is the rotation of the end effector frame O_e in relation to the base frame O_b:

$$\boldsymbol{R_e^b} = \begin{bmatrix} \boldsymbol{n} & \boldsymbol{s} & \boldsymbol{a} \end{bmatrix} = \begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix} \tag{3.1}$$

where the base frame is the frame attached to the last passive set up joint of the PSM. The column vectors of Equation 3.1 correspond to the unit vectors of the coordinate frame of the end effector shown in Figure 3.1. The $\boldsymbol{n}$, $\boldsymbol{s}$, $\boldsymbol{a}$ notation is used for an end effector with a gripper. The z axis becomes $\boldsymbol{a}$, known as the *approach* direction, and the y axis becomes $\boldsymbol{s}$, the *sliding* plane of the gripper. This is the direction the gripper opens, normal to $\boldsymbol{a}$. Then $\boldsymbol{n}$, is chosen orthogonal to the other two, completing the right handed coordinate frame.

In order to control the PSM tool orientation from the Leap Motion controller and the hand, a rotation matrix must be created to command the tool to the desired orientation. The orientation of a rigid body can also be represented by a combination of the three Euler angles about the three orthogonal axes of a coordinate system. The leap motion API provides the angles of the hand using the *Roll-Pitch-Yaw angle* (RPY) convention, typically used in the aeronautical field for a heading of an aircraft. The angles represent rotations with respect to a frame attached to the

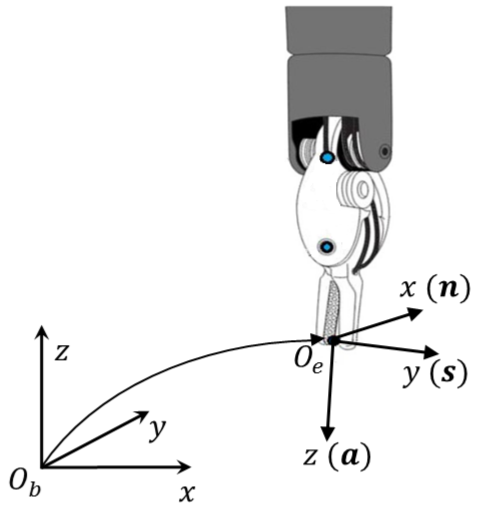
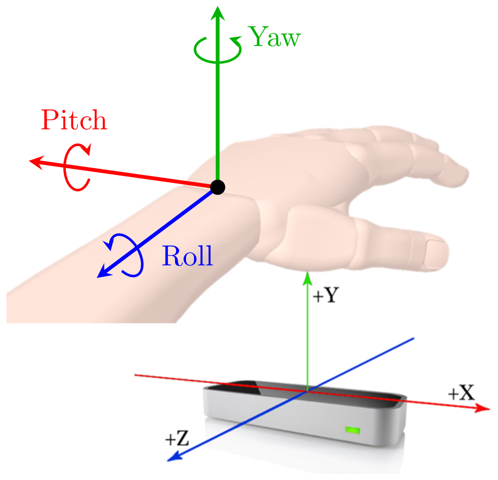
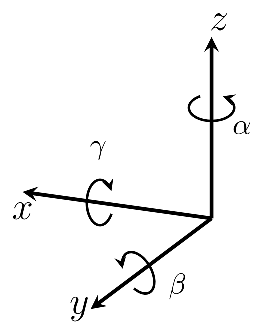


Figure 3.1: End effector coordinate frame

centre of mass of the object. Leap motion defines the roll angle about the z axis, which is the roll of the wrist, i.e., supination and pronation. The pitch angle is the rotation about the x axis, or flexion and extension of the wrist. Finally, the yaw angle is about the y axis, or wrist adduction and abduction. These angles and the base coordinate frame of the Leap Motion controller are shown in Figure 3.2a.



(a) Leap motion rotation angles



(b) converted coordinate system

Figure 3.2: Coordinate systems

The coordinate system in the Leap API is defined as z towards the user, x to the right, and y vertical from the sensor. However, the PSM uses a different coordinate system, with z vertical,

x to the left, and y towards the front of the robot. For ease of calculations, these rotation angles are converted to rotations about that coordinate system, as shown in Figure 3.2b. Yaw becomes a counter-clockwise rotation of α about the z axis, roll, a counter-clockwise rotation of β about the y axis, and pitch, a counter-clockwise rotation of γ about the x axis. The rotation matrices representing the individual rotations are given by:

$$\boldsymbol{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

$$\boldsymbol{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{3.3}$$

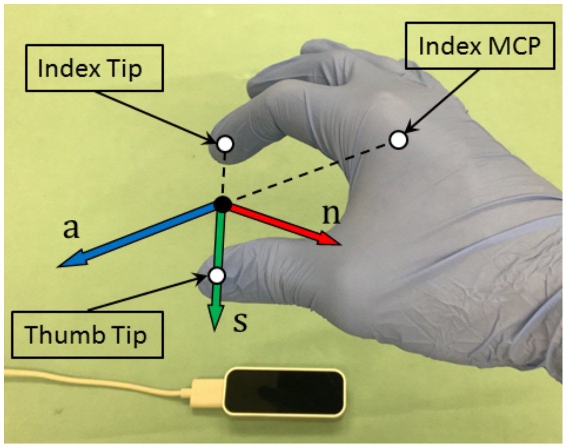
$$\boldsymbol{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \tag{3.4}$$

The three rotations can be used to describe the orientation of the hand with respect to the fixed coordinate system of the leap motion controller (with the axes swapped to match the PSM). Multiplication of the three rotation matrices gives a single rotation matrix that fully describes the orientation of the hand:

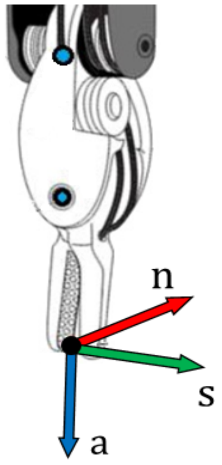
$$\begin{aligned} \boldsymbol{R}_{hand}(\alpha,\beta,\gamma) &= \boldsymbol{R}_z(\alpha)\boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\gamma) \\ &= \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix} \end{aligned} \tag{3.5}$$

This rotation matrix now describes the orientation of the hand with respect to the Leap Motion controller, and is used to orient the tool with respect to the base frame of the PSM.

This representation of the hand orientation was briefly tested to control the orientation of the tool. It was found that the tool did not match the user's motions very accurately when attempting to pick up an object. For effective teleoperation, controlling the tools should feel as natural and intuitive as possible. The RPY orientation angles were not adequate for representing the desired orientation when attempting to pick up an object. These Euler angles did not take into account the individual motion of the fingers when grasping, as the fingers add dexterity over the overall orientation of the hand. Analyzing a human hand when picking up an object with thumb and index finger, the desired approach vector, **a**, can be described by a vector from the metacarpophalangeal (MCP) joint of the index finger to the position between the thumb and index finger tips (Figure 3.3), as follows:



(a) Hand coordinate frame



(b) End effector frame

Figure 3.3: End effector coordinate system

$$\boldsymbol{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} (indexTip.x + thumbTip.x)/2 - MCP.x \\ (indexTip.y + thumbTip.y)/2 - MCP.y \\ (indexTip.z + thumbTip.z)/2 - MCP.z \end{bmatrix} \tag{3.6}$$

The **s** direction is then generated by taking the vector from the index tip to the thumb tip, and projecting it onto the plane normal to the generated $\boldsymbol{a}$ to make it orthogonal. This vector controls the direction of opening of the tool, using the finger and thumb as either side of the gripper. The equation for projecting a vector onto a plane is given by:

$$proj_{Plane} = \boldsymbol{u} - proj_{\boldsymbol{n}}(\boldsymbol{u}) = \boldsymbol{u} - \frac{\boldsymbol{u} \cdot \boldsymbol{n}}{||\boldsymbol{n}||^2}\boldsymbol{n} \tag{3.7}$$

Where $\boldsymbol{u}$ is the vector to be projected, and $\boldsymbol{n}$ is the normal vector to the plane. The plane that is to be projected onto is the plane normal to the **a** vector, and the vector to be projected is from the thumb tip to the index tip. The resulting equation is given as:

$$\boldsymbol{s} = \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} = (thumbTip - indexTip) - \frac{(thumbTip - indexTip) \cdot \boldsymbol{a}}{||\boldsymbol{a}||^2}\boldsymbol{a} \tag{3.8}$$

The final axis, x, or **n** is then generated by the cross-product between the $\boldsymbol{s}$ and $\boldsymbol{a}$ axes.

$$\boldsymbol{n} = \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \boldsymbol{s} \times \boldsymbol{a} \tag{3.9}$$

These three orthogonal vectors are then normalized, and used to create the right handed rotation matrix $\boldsymbol{R}_{hand}$.

$$\boldsymbol{R}_{hand} = \begin{bmatrix} \boldsymbol{n} & \boldsymbol{s} & \boldsymbol{a} \end{bmatrix} = \begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix} \tag{3.10}$$

This rotation matrix is now used to command the tool to the desired orientation, with respect to the base frame of the PSM. This base frame is not fixed however, and is attached to the final joint of the passive set up joints. These 5 additional passive set up joints are used to position the arm before surgery (Figure 3.4). There are also 3 passive set up joints on the ECM arm. When these passive joints are moved, an additional rotation is introduced between the tools and the frame of reference that the surgeon is placed in. Therefore, an additional rotation matrix is needed to account for the rotation of the passive joints between the ECM and the base frame of the PSM.

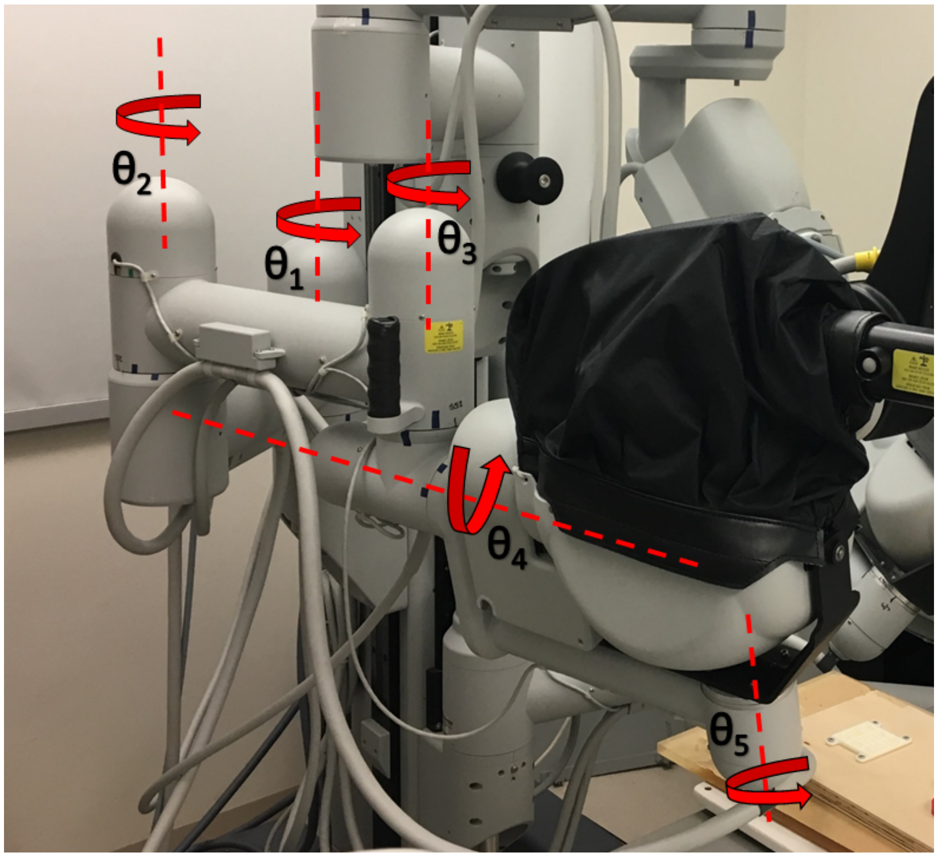


Figure 3.4: PSM set up joints showing axes of rotation

The set up joints are not interfaced with the dVRK, so the da Vinci API, available for read only research with the classical system is used to read out the joint angles of the set up joints for the PSM and the ECM. Using these angles, we are able to get the orientation of the PSM with respect to a defined world frame $\boldsymbol{R}_{PSM}$, as well as the ECM with respect to the world frame, $\boldsymbol{R}_{ECM}$. The rotation matrix can be computed as series of rotations about the z and y axes, as follows:

$$\boldsymbol{R}_{PSM} = \boldsymbol{R}_z(\theta_1)\boldsymbol{R}_z(\theta_2)\boldsymbol{R}_z(\theta_3)\boldsymbol{R}_y(\theta_4)\boldsymbol{R}_z(\theta_5) \tag{3.11}$$

$$\boldsymbol{R}_{ECM} = \boldsymbol{R}_z(\theta_1)\boldsymbol{R}_z(\theta_2)\boldsymbol{R}_z(\theta_3) \tag{3.12}$$

where $\boldsymbol{R}_z$ refers to Equation 3.2, and $\boldsymbol{R}_y$ refers to Equation 3.3. The rotation from ECM to PSM can then be defined as:

$$^{ECM}\boldsymbol{R}_{PSM} = (\boldsymbol{R}_{ECM})^{-1} \cdot \boldsymbol{R}_{PSM} \tag{3.13}$$

The rotation from ECM to PSM is then used to pre multiply the rotation matrix created from the Leap Motion controller:

$$\boldsymbol{R}_{final} = {}^{ECM}\boldsymbol{R}_{PSM} \cdot \boldsymbol{R}_{hand} \tag{3.14}$$

This final rotation matrix is then used as the command to send to orient the PSM tool. This places the tool in the correct orientation, so when viewed through the endoscope it matches the user's hand orientation. The setup joints are positioned once before surgery, and not moved during operation, so this rotation matrix is hard coded, and only changed when the external setup needs adjusting. This matrix only accounts for the passive joints of the ECM, when in home position. The camera arm is also controllable, and when moved, an additional rotation matrix is needed to account for this rotation, as addressed in Chapter 4.

3.2.2 Position

Position control of the PSM is accomplished by controlling the position of the wrist of the tool, i.e., controlled by the first three outer joints of the PSM, while the remaining joints only control orientation. This wrist position of the tool is controlled by the surgeon's wrist position tracked by the Leap Motion controller. This position was chosen as it is the point of rotation for the hand and remains still while the hand is rotated. The PSM arms are positioned at the beginning of a procedure, and therefore there is no consistent starting position for the robotic tools. For this reason, control of the tool position is not done globally, but instead it is computed relative to this initial position of the tool wrist.

The Leap Motion controller is constantly tracking the position of the hand as a Cartesian coordinate point: $\boldsymbol{p}_{wrist} = \left[p_{wrist_x}\ p_{wrist_y}\ p_{wrist_z}\right]^T$ An initial wrist position is saved at the beginning of operation, and used in combination with the position in subsequent frames to compute the change in hand position for the current frame:

$$\boldsymbol{v}_{wristRelative} = \boldsymbol{p}_{wristCurrent} - \boldsymbol{p}_{wristInitial} \tag{3.15}$$

This vector is then transformed by multiplication with the rotation matrix ${}^{ECM}\boldsymbol{R}_{PSM}$ in Equation 3.13 to align the movements of the hand with the view from the endoscope.

$$\begin{aligned}\boldsymbol{v}'_{wristRelative} &= {}^{ECM}\boldsymbol{R}_{PSM} \cdot \boldsymbol{v}_{wristRelative} \\ &= \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{wristRelative_x} \\ \boldsymbol{v}_{wristRelative_y} \\ \boldsymbol{v}_{wristRelative_z} \end{bmatrix}\end{aligned} \tag{3.16}$$

The relative vector is used to compute the new desired position of the PSM tool wrist, relative to the position saved at the beginning of operation.

$$\boldsymbol{p}_{PSM} = \boldsymbol{p}_{PSMinitial} + (\boldsymbol{v}'_{wristRelative} \cdot scaleFactor) \tag{3.17}$$

The scaling factor is a variable that can be changed depending on the desired precision. The default value is 0.25, meaning that the user's translational movements are scaled down by a factor of 4. The position and orientation are combined into a pose command, and sent to the PSM through ROS to command the tool to the desired pose. Section 3.2.4 details the full operation, including when and how the initial positions are saved, and use of the foot pedals.

3.2.3 Gripping Angle

The 7th and last DOF to control is the opening and closing of the tool, used for grasping objects or cutting. With the dVRK, the gripping angle can be controlled by publishing the desired angle, with a float data type on the topic `/dvrk/PSM/set_jaw_angle`. This angle can be controlled using the pinching motion of the index finger and the thumb tips. When a person picks up a small object with their hand, the most natural way is to pinch the object between the index finger and thumb. Therefore the positions of these two fingers are used to control the last degree of freedom of the tool. Two vectors are created: from the index MCP joint to the index tip, and from the index MCP joint to the thumb tip, shown in Figure 3.5. The smallest angle between the two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is given by:

$$\cos(\alpha) = \frac{\boldsymbol{a} \cdot \boldsymbol{b}}{|\boldsymbol{a}| \cdot |\boldsymbol{b}|} \tag{3.18}$$

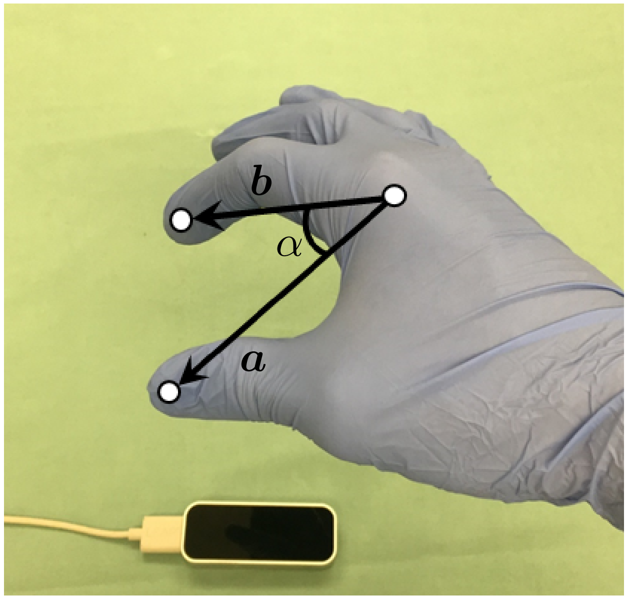


Figure 3.5: Gripping angle vectors

The values for α range from 5 degrees when pinching, to greater than 100 degrees when the fingers are spread apart as far as possible. This angle is then scaled to the appropriate value for the end effector. The minimum and maximum angles for the end effector are -20 and +80 degrees respectively. The negative joint angle commands allow for the gripper to apply more force when pinching, necessary for grasping small objects. Therefore there are three positions of the end effector to be scaled between:

- *fully open*, defined by the mechanical limit of the manipulator, +80 degrees
- *closed but not tight*, 0 degrees
- *closed tightly*, defined by the minimum limit for the gripper, -20 degrees

The angles of α to correspond to these three positions were determined experimentally. When pinching the thumb and index finger together, the average angle α was found to be 5 degrees, so this is chosen as the closed tightly position. The fully open position was chosen as 70 degrees, as a comfortable angle to provide the maximum angle of the tool without having to strain the fingers to open further. A value of 20 degrees was then determined to be a suitable angle for the closed but not tight position. The values are then scaled linearly between these positions to the desired angle for the end effector. This is summarized in Table 3.1. This results in the tool closing when the users fingers are not quite pinched together fully, and additional closing provides a negative angle to the tool.

Table 3.1: Scaling of gripping angle to manipulator

	Fully open	Closed	Closed Tightly
Hand (α)	70	20	5
Manipulator	80	0	-20

3.2.4 Full Teleoperation

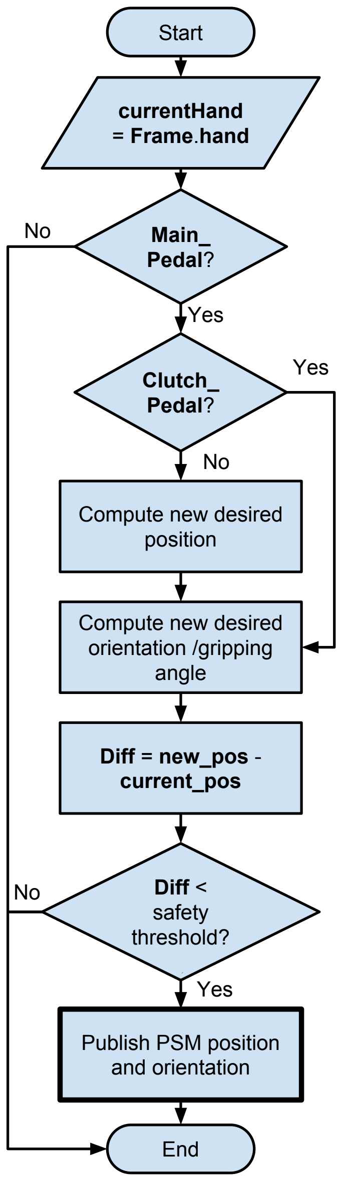
The system works in much the same way as the da Vinci system currently does; the surgeon's hands are used to control the position of the PSM tools, and foot pedals are used to increase functionality. The first pedal, pressed by the right foot, is used as a safety or dead man's switch, meaning the pedal must be pressed for the system to respond to any movements. If the pedal is released, teleoperation is stopped. To begin, the user places their hands in the workspace above the Leap Motion controller, and once ready, presses the pedal to engage and initialize the orientation of the tools. Once the pedal is pressed, the current Cartesian coordinates of the PSM tool tip are saved, as well as the current position of the surgeon's wrist. These initial positions are then used in Equation 3.17 to compute the desired position for the tool during operation.

The tools are initialized with the orientation of the hands, saved at the time of the footpedal press. This initial orientation is sent to the tools as the final orientation for a trajectory generator. This moves the tool from the current orientation to the desired orientation smoothly, and ensures the hand and tool orientations match before beginning teleoperation. The initialization does not move the position of the tool wrist, only the orientation. As soon as the goal is reached, teleoperation begins, and subsequent commands are then sent directly to position the joints to reach the desired point immediately.

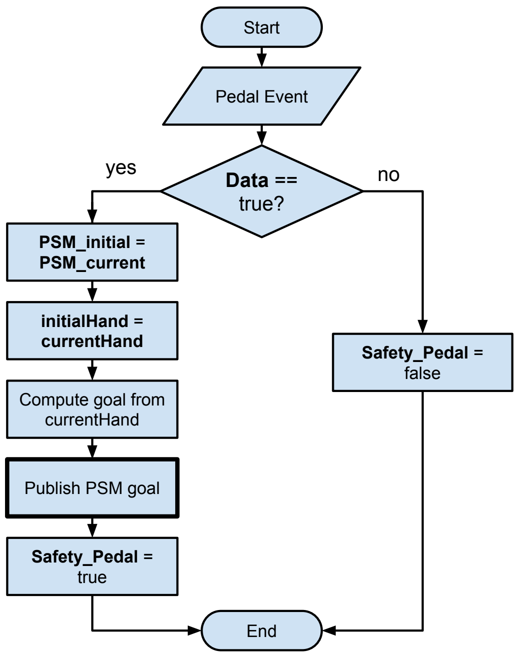
There is an additional clutch pedal used for repositioning the hands during operation. While the clutch pedal is pressed, the position is not updated, allowing the user to reposition their hands in the workspace without controlling the slaves. This allows the surgeon to move the tools farther than possible in the confined workspace for the hands.

There are two main ROS nodes involved in performing teleoperation with the Leap Motion controller. The first node connects to the Leap Motion sensor, and publishes the data of the hand on a custom ROS message. The custom message includes the orientation angles of the hand, the wrist and palm position, the position of the finger joints, as well as a time-stamp and frame ID number for error checking. The leap motion controller is polled at 100 Hz to obtain a new frame object that includes the hand data. The second node, *leap_dvrk_bridge*, subscribes to the hand data from the Leap motion controller, the current position of the PSMs, as well as the footpedals. Within the second node is where all the computation is done. There are several callback functions within the leap_bridge node that are run continuously, used to update variables including the hand structures, the pose of the PSMs, and the status of pedals. The main callback: *hand_callback*, updates a hand object on every frame, and then computes the pose command sent to the PSM depending on if pedals are pressed. It then checks the command to be sent with the current pose, and if the difference in the orientation angle or position is under a defined threshold, the pose command will be sent. This is a safety to ensure that the tools never move in an undesirable way, in case of a tracking error. If the new command is over the threshold, then the command will not be sent, and will wait until the next frame that is close enough to the current pose of the tool. The threshold is not too small, such that the hand can be moved back relatively close to the last sent position and tracking resumes.

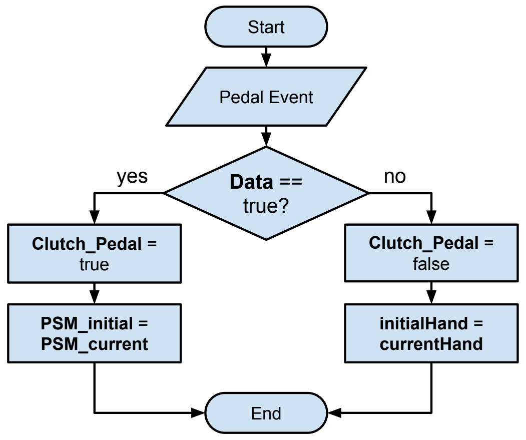
Additional callbacks are used for responding to the pedals. The callbacks are called only on a change of event, (i.e., when the pedal is pressed or released). A flag is set to true when the pedal is pressed, and false when the pedal is released. When the flag is changed, it alters the functionality of the hand callback. If the safety pedal flag is not set, the hand callback only updates the current hand variable *CurrentHand*, and then returns, not sending a command to the PSM. If the clutch pedal flag is set, the orientation command is still sent, but the position is not updated. When the clutch pedal is released, the initial position of the hand is updated to move relative to current position. A flowchart of the system can be seen in Figure 3.6. Not shown are the callbacks which respond to the pose of the PSMs, as they simply update a single structure holding the current pose of the PSM tool.


Start
currentHand = Frame.hand
Main_ Pedal?
No
Yes
Clutch_ Pedal?
Yes
No
Compute new desired position
Compute new desired orientation /gripping angle
Diff = new_pos - current_pos
Diff < safety threshold?
No
Yes
Publish PSM position and orientation
End

(a) Hand Callback



(b) Safety Pedal Callback



(c) Clutch Pedal Callback

Figure 3.6: Callback functions of ROS nodes

A boundary box was also set up to indicate to the user when they are outside of the designated tracking area. As the user's hands move to the edge of the boundary, an auditory alert is sent to inform them to move back to the center of the workspace. Section 3.3 gives recommendations for the size of the workspace to be used during surgery.

3.2.5 Validation

To test the performance of the system initially, a pick and place task was performed with three subjects. The task involves picking up the different coloured tacks, and placing them into the corresponding coloured bins as seen in Figure 3.7. There are three of each colour, for a total of 9 tacks. Two out of three users were able to successfully pick up and place all 9 tacks into the respective containers without any drops. One user had 2 drops, but was able to successfully pick back up and complete the task. Although this was promising, the hand does not have to rotate very far to complete the task, and is always in an orientation suitable for tracking from the controller. With more complex tasks, as the hand is rotated into different orientations, the tracking of the fingers degrades significantly due to occlusions, causing unwanted movements. The next section outlines attempts to address this problem by integration of multiple sensors from different viewpoints. Another thing to consider, is this task was performed with the Leap Motion controller situated directly in front of the robot. The endoscopic camera was not used for this task, and was integrated with a head mounted display, outlined in Chapter 4.

3.3 Occlusion/Multiple Controllers

As with all optical tracking systems, the Leap Motion controller suffers from occlusion issues, or line of sight. As the hand is rotated into certain orientations, the tracking performance degrades due to fingers being occluded from the line of sight of the camera. This can happen due to occlusion from the palm, or from the other fingers. In addition to occlusion, aliasing also occurs when there is an object directly behind the fingers, and the finger position can not be determined accurately. For this reason, there should be sufficient space above the hand so that the sensor does not detect anything directly behind it. It is also for this reason that the controller needs to be placed so that

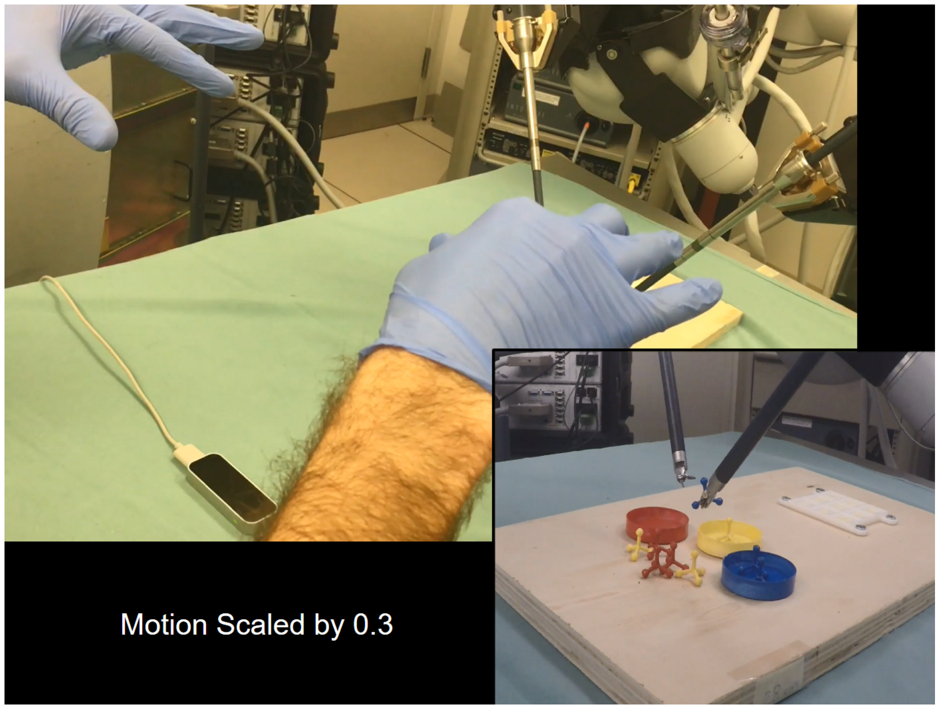
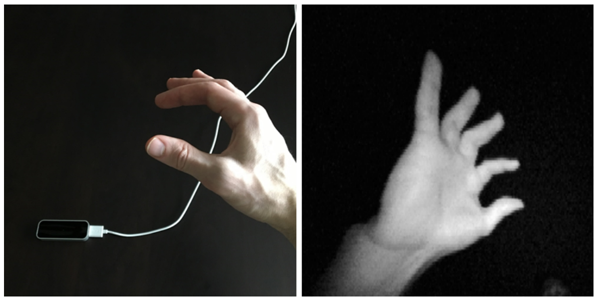


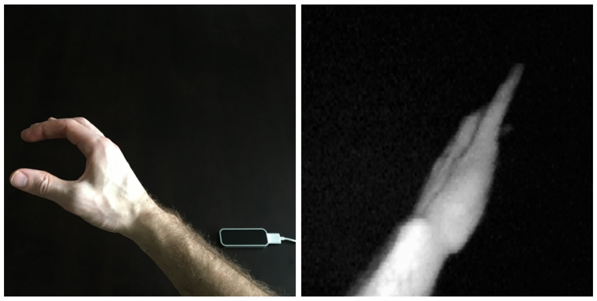
Figure 3.7: Pick and place task with Leap Motion

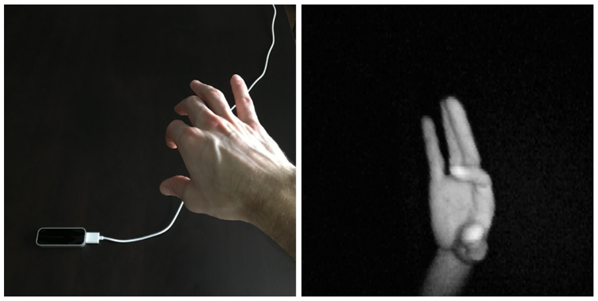
the user's body and or head do not interfere with the tracking.

The main issues with occlusion or aliasing happen due to the roll of the hand, since it has a much bigger range than pitch and yaw do for the human wrist. Figure 3.8 shows the images taken from the Leap Motion visualizer in two orientations, varying the horizontal translation. The left images show the physical position of the hand with respect to the sensor, while the right images show the infrared image taken from the sensor. Figures 3.8a and 3.8b show the hand rotated counter-clockwise, translated in the positive and negative x axis respectively. When to the right of the sensor, the thumb and finger tip are seen clearly, but when to the left of the sensor the finger tips are occluded by the palm of the hand. A similar scenario can be seen when the hand is rotated clockwise. When translated in the positive x axis, the palm is directly behind the finger tips causing inaccurate tracking. However, in this same orientation when translated to the left, the finger tips are seen clearly.

In order to improve tracking, multiple controllers were used to have multiple viewpoints on the hand, overcoming the occlusion issues. When the hand moves into an orientation not tracked well by one sensor, the system switches to another sensor with a different view point. To use multiple


(a)


(b)


(c)

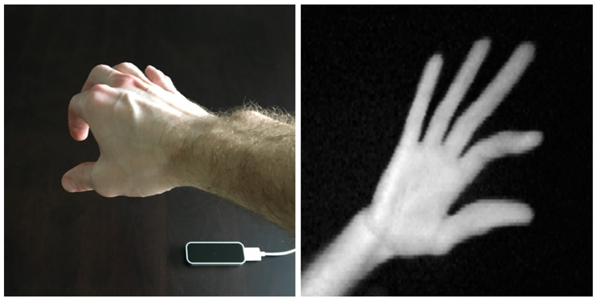

(d)

Figure 3.8: Images of hand from Leap Motion controller in various poses. The right images come from the Leap Motion visualizer and left images corresponds to the real hand. (a) counter-clockwise translated right, (b) counter-clockwise translated left; (c) clockwise translated right; (d) clockwise translated left

sensors, a secondary Leap Motion controller was connected using a virtual machine installed on the PC. This is due to limitations with the Leap Motion software, where only one device can be connected at one time. The virtual machine is referred to as the guest operating system (OS), and the main computer referred to as the host OS. In order to synchronize the data with the controllers on the guest and host OS, ROS is used to communicate between the two. The data from the secondary sensor are published from the guest, and subscribed to by the host. The data are sent in the same custom message, `Leap_msgs:Hands`, defined previously in Section 2.3.3.1, which publishes all the joint positions, RPY angles, and a timestamp. The data are synchronized between the two sensors on the host OS.

Within this section, a coordinate system is defined as x to the right, y away from the user, and z increasing vertically up. This global coordinate system is a right handed coordinate frame that is intuitive to use; instead of having z point towards the user, it was oriented vertically, which is more common.

3.3.1 First Attempt

The first attempt to combine multiple Leap Motion controllers was to use relative orientation, so the second controller would not need to be calibrated into the same global frame. This would allow the positioning of the second sensor at any position and orientation. As the hand moved into an orientation that would be tracked poorly by one controller, the second controller would take over. The switch was done by saving the orientation right as it switches, then computing the new orientation relative to that one. This method seemed to work well initially, however, after many switches, small errors between the two controllers would eventually add up, causing drift between the hand orientation and the tool orientation. The orientation of the tool was no longer related to a global reference frame, which defeated the purpose of having the global frame of reference, useful with vision based tracking. It was decided that a global reference frame would be needed, and the two sensors needed to be calibrated so that the position and orientation being reported was relative to the same fixed frame.

Multiple Leap Motion controllers have been used before, for combining the tracking of two sensors from different view points [90]. A simple calibration is done to obtain the orientation of one sensor relative to the other, using the tracked hand data reported from each sensor. This allows for a fast calibration, placing the sensor in any position. A similar approach was done here to attempt to calibrate the sensors. The first sensor was placed flat on a table, and the secondary controller was held to the side. The sensors were placed in the same x–z plane, so that the rotation of the side sensor about the y axis places the center of the two workspaces at the same point. A calibration technique was used to generate the transformation needed to align the controllers coordinate systems. This technique was used to determine if it was feasible to combine multiple Leap Motion controllers for this application. There was no set position defined, allowing the reposition of the sensor and recalibration. Figure 3.9 shows the setup used for this calibration, with the global coordinate system centred on the base controller. $\boldsymbol{O}_b$ and $\boldsymbol{O}_s$ are the origins of the base sensor and side sensor respectively, and $\boldsymbol{O}_h$ is the origin of the tracked hand, centred at the wrist. ${}^{B}\boldsymbol{T}_h$ and ${}^{S}\boldsymbol{T}_h$ are the pose transformations of the hand, from the base and side sensor respectively. These pose transforms are 4×4 matrices for a compact representation of a translation

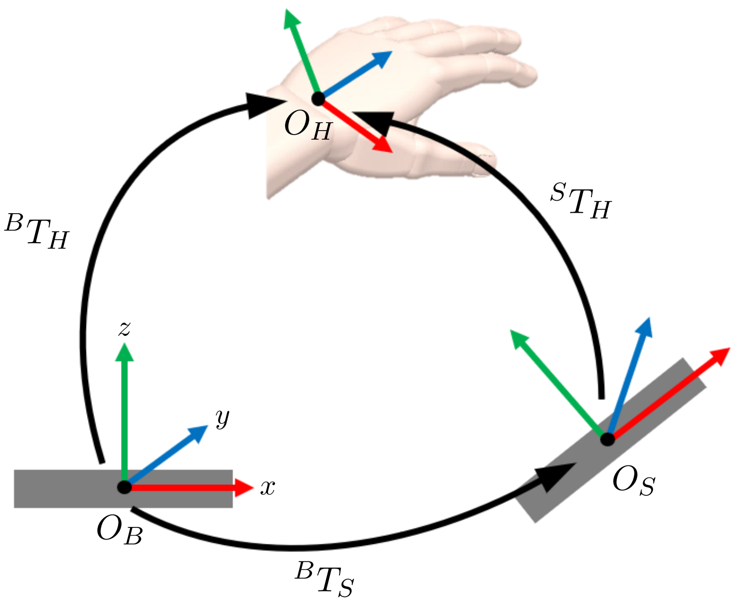


Figure 3.9: Coordinate transformation for secondary Leap Motion sensor

and a rotation as follows:

$$\begin{pmatrix} \begin{bmatrix} & & \\ & \mathrm{R} & \\ & & \end{bmatrix} & \begin{matrix} \boldsymbol{p}_x \\ \boldsymbol{p}_y \\ \boldsymbol{p}_z \end{matrix} \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix}$$

Just as with 3×3 rotation matrices used previously, these can be multiplied together to convert coordinates into a new coordinate frame. To fuse the data from the two sensors, the transformation is needed from the base to the side controller ${}^{B}\boldsymbol{T}_S$. This transform is used to take the data from the secondary side sensor, and transform it into the global coordinate frame. This is obtained by:

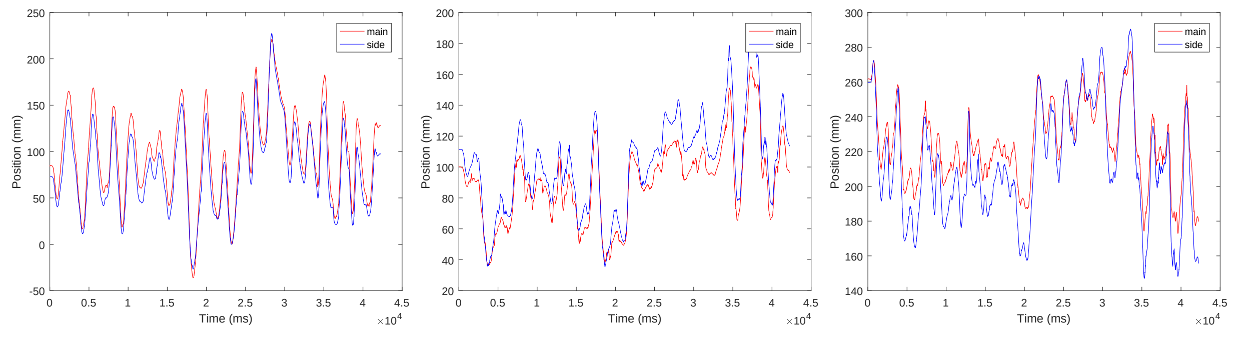
$$ {}^{B}\boldsymbol{T}_S = {}^{S}\boldsymbol{T}_H \cdot {}^{B}\boldsymbol{T}_H^{-1} \tag{3.19}$$

where this calibration matrix now gives the pose of the side sensor in terms of the base sensor. The calibration technique is as follows: hold hand still in the middle of the workspace, and obtain

10 measurements from each leap controller as the hand is translated slowly around the workspace. The position of the wrist and RPY Euler angles are used to create an an array of 10 transformation matrices from both sensors $[{}^{B}\boldsymbol{T}_{H1}, {}^{B}\boldsymbol{T}_{H2}, ..., {}^{B}\boldsymbol{T}_{H10}]$, and $[{}^{S}\boldsymbol{T}_{H1}, {}^{S}\boldsymbol{T}_{H2}, ..., {}^{S}\boldsymbol{T}_{H10}]$. Then Equation 3.19 is used to compute the calibration matrices: $[{}^{B}\boldsymbol{T}_{S1}, {}^{B}\boldsymbol{T}_{S2}, ..., {}^{B}\boldsymbol{T}_{S10}]$. The Euler angles, and Cartesian positions of these matrices are extracted, and averaged to reconstruct the final transformation matrix. This matrix is saved, and then used during operation to align the hand data in the same coordinate system:

$$\boldsymbol{T}_{converted} = {}^{B}\boldsymbol{T}_{S} \cdot {}^{S}\boldsymbol{T}_{H} \tag{3.20}$$

After performing the calibration, the hand was moved around the workspace, keeping the orientation of the hand fairly still so that the fingers were not occluded by either sensor. The pose of the hand was tracked from both sensors, (transforming the second sensor data). The position and orientation from each sensor were then extracted from this transformation matrix. The position of the index finger tip compared between the two sensors is plotted in Figure 3.10.



(a) (b) (c)

Figure 3.10: Comparing Position of Multi Sensor Setup. (a) x axis; (b) y axis, (c) z axis

The calibration worked fairly well, but was not perfect, causing the error to get large as the hand was moved away from where it was calibrated. The 3D position error in millimetres between the original sensor and the side sensor is plotted as a function of time in Figure 3.11. The mean error between the two sensors was 10.2 mm, with the maximum error being 28.5 mm. This shows that it is possible to combine multiple sensors, and within a neutral range of roll angles the hands

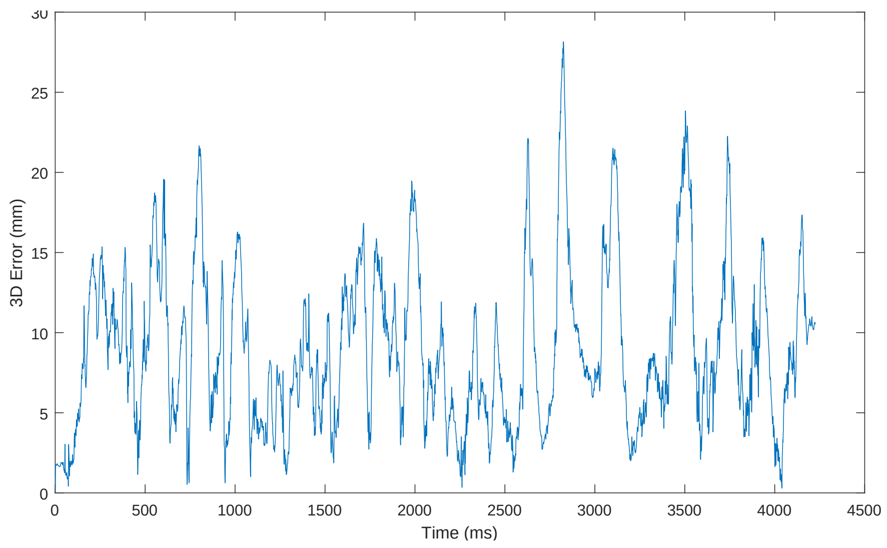


Figure 3.11: 3D position error of multi sensor setup

are tracked fairly well. With a hardcoded setup it was assumed that the calibration would be much better, so the position of the sensor needed to be determined to best maximize the tracking area. In order to switch between the controllers, the point at which to switch, and where to place to controllers needed to be defined. This was done by assessing the accuracy of tracking within the entire workspace, and determining where exactly the tracking issues arise.

3.3.2 Tracking Analysis

To assess the tracking accuracy of the leap motion controller, and where exactly the tracking issues arise, the tracking of the Leap Motion controller was compared with an the Aurora®electromagnetic (EM) tracking system (Northern Digital, Inc.) [91]. EM tracking does not have the same issues with occlusion as an optical system, therefore the EM trackers can be used to compare to the Leap Motion sensor. The Aurora system consists of an electric field generator, and 6 DOF sensors. These sensors have a suitable accuracy and precision for this application, with a position accuracy error of 0.48 mm RMS, precision error 0.30 mm RMS, and trueness of 0.4 mm RMS (ISO 5725-1 (Accuracy (Trueness and Precision) of Measurement Methods and Results)). The orientation error for a 6DOF sensor is: accuracy 0.3 deg RMS, precision: 0.2 deg RMS, and trueness 0.25 deg RMS [92]. The Aurora system is used as the gold standard to assess the accuracy of the Leap Motion controller. Since the occlusion issues of the Leap Motion controller cause much greater

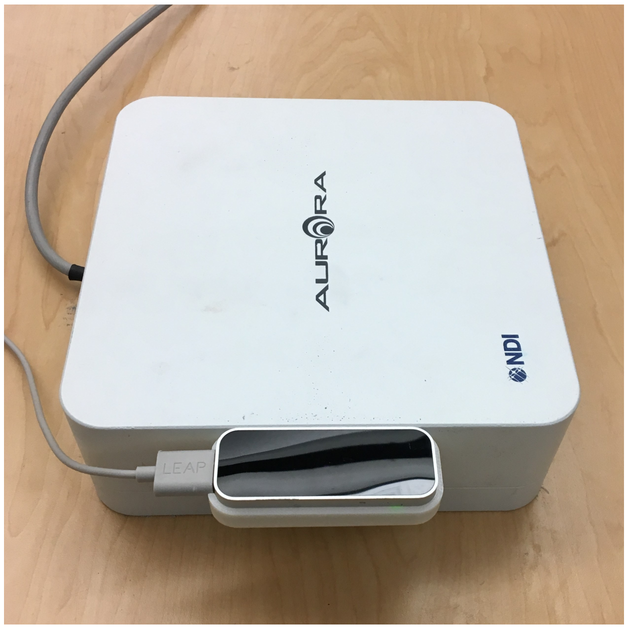
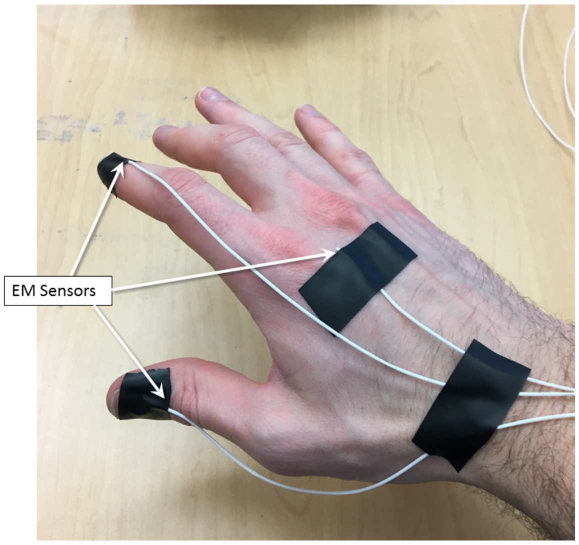


Figure 3.12: Leap Motion controller mounted to Aurora electromagnetic field generator

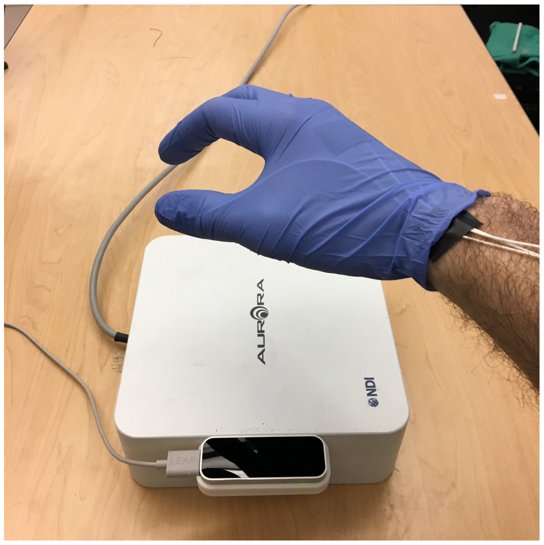
error than the accuracy of the sensors, it is suitable to use it to compare the performance of the Leap Motion controller, and determine where the tracking issues occur.

The system was used to track the three joints of interest, the index finger tip, the thumb tip, and the index MCP joint. The frame rates of the two systems were synchronized to get a data set from the Leap Motion controller and the Aurora that corresponds to the same moment in time. To compare the two sets of data, first the coordinate systems of the two systems were aligned. The same right handed coordinate system is defined as in Section 3.3.1, with the origin at the center of the Leap Motion controller. The original origin of the EM tracking system is in the center of the upper face of the generator. A custom 3D printed mount was designed to align the x–y, and y–z planes of the two systems, leaving only a translation in the y direction (Figure 3.12). An offset in the y direction was applied to the EM data to fully align the two coordinate frames.

To track the finger positions, three 6 DOF sensors were placed on the joints of interest. They were placed on the fingers and taped down, to ensure there was no movement relative to the finger when the hand was moving. Figure 3.13 shows the placement of the trackers on the hand. A surgical glove was used to cover the EM sensors and tape to ensure consistent tracking. Only the right hand was analysed, and any conclusions were then mirrored for the left hand.



(a) EM sensor placement



(b) Glove covering EM sensors

Figure 3.13: Electromagnetic sensors on fingers

The Leap Motion controller tracks the position of the fingers joints at their center. Since the EM sensors are placed on the surface of the skin of the hand, there is a slight error in position when comparing the two. To account for this, a rigid offset from the sensor to the leap joint position was calculated, and used to position the Aurora data at the same point as the Leap Motion controller data during tracking. By knowing the position of the Leap Motion joint with respect to the EM sensors frame of reference, this offset can be used to align the data when the hand is moved into a new pose. Figure 3.14 illustrates how this is accomplished. Consider P, the position of the joint as reported by the Leap Motion controller, where $\boldsymbol{p}^0$ is the vector coordinate of P with respect to the global coordinate frame O_0. Considering another frame O_1, the Aurora EM sensor frame, where the vector $\boldsymbol{p}_1^0$ describes the frame's position with respect to the reference frame, and $\boldsymbol{R}_1^0$ represents the orientation of the EM sensor with respect to the reference frame, then the point P can be expressed with respect to frame 0 as:

$$\boldsymbol{p}^0 = \boldsymbol{p}_1^0 + \boldsymbol{R}_1^0\, \boldsymbol{p}^1 \tag{3.21}$$

This can be rearranged to solve for $\boldsymbol{p}^1$ as:

$$\begin{aligned} \boldsymbol{p}^1 &= -\boldsymbol{R}_1^{0T}\boldsymbol{p}_1^0 + \boldsymbol{R}_1^{0T}\boldsymbol{p}^0 \\ &= \boldsymbol{R}_0^1(\boldsymbol{p}^0 - \boldsymbol{p}_1^0) \end{aligned} \tag{3.22}$$

This represents the offset of the leap joint position with respect to the EM tracker frame of reference. When the hand is moved into a new orientation, this offset can be used to reposition the Aurora data onto the same point as the Leap sensor data.

A calibration routine was performed in order to obtain this offset, where the hand was held still above the sensors while in a pose that was tracked accurately by the Leap Motion controller. Ten frames of data from both the Leap Motion controller and the Aurora tracking system were recorded, and Equation 3.22 was used to compute the vector offset of each frame. The mean value of this vector was saved. It was then used to compare this converted position with the Leap position as the hand was moved to different poses around the workspace of the Leap Motion controller. The error between these two positions indicated the performance of the tracking of the sensor.

After calibration, and the offset saved, the hand was moved around the workspace above the two systems, in motions that would be seen when operating with the robotic system. The hand was oriented in a pinching gesture, and translated around the workspace, while simultaneously being rotated through the wrist's range of motion. This allowed the performance of the leap to be assessed in a real world test, and included any occlusions that would arise during surgery, and what position and orientation the hand was in when they occurred. For every frame of data, the rotation matrix output from the EM sensor, $\boldsymbol{R}_1^0$ was used in combination with the offset $\boldsymbol{p}_1$, calculated in the calibration procedure to recentre the aurora data onto the leap data. The inferred position of the joint using the position of aurora tracker becomes:

$$\boldsymbol{p}^{0\prime} = \boldsymbol{p}_1^0 + \boldsymbol{R}_1^0\,\boldsymbol{p}^1 \tag{3.23}$$

This was computed for all three of the joints of interest for every frame, and then analysed in MATLAB comparing the position of $\boldsymbol{p}^{0\prime}$ and $\boldsymbol{p}^0$. Figure 3.15 shows the comparison of the position

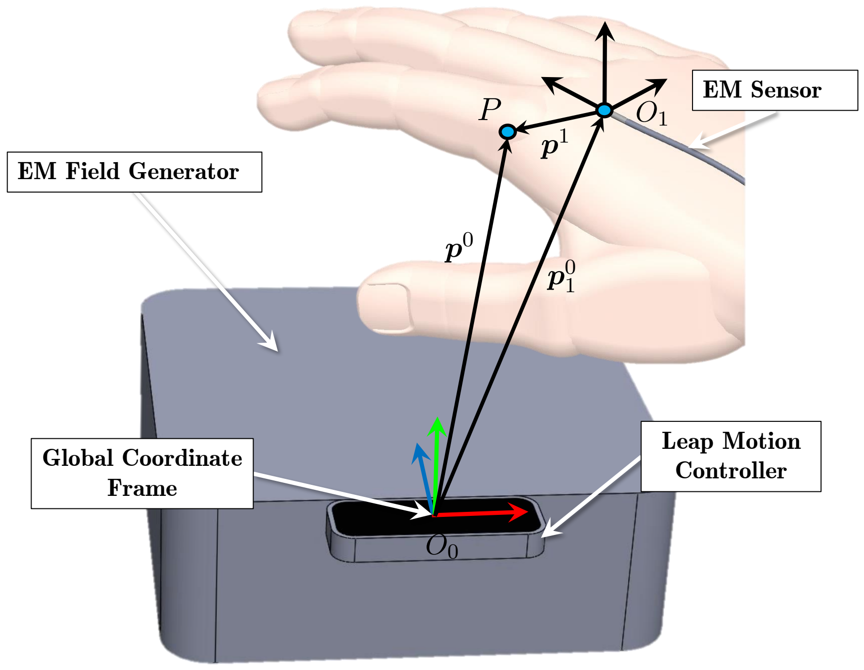


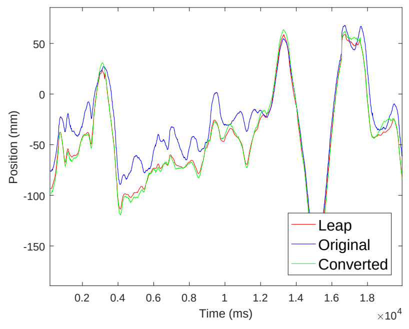
Figure 3.14: EM tracker calibration

data in all three axes for the MCP joint over a short period. This joint is shown, since of the three EM sensors, this one was placed the farthest from the joint of interest, making it easier to visualize the correction. The blue line represents the raw EM tracking data, the red line the Leap Motion data, and the green line is the converted data from the EM tracker. As seen in the graphs, the green line now tracks the red line much better than the original blue data. The same can be seen for the index tip and the thumb tip position.

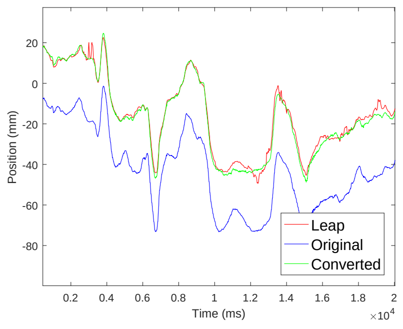
The 3D position error for the three joints of interest was calculated between the Leap Motion controller position and the converted EM position:

$$Error = \boldsymbol{p}^{0'} - \boldsymbol{p}^0 \tag{3.24}$$

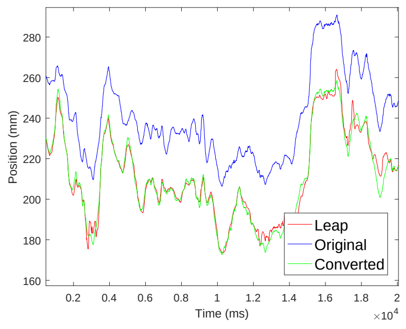
This error indicates how well the leap motion controller was performing. Areas that have a higher error indicate the area of the workspace where the tracking is not as accurate. Figure 3.16 shows the 3D positional error of the three joints plotted as a function of time. Highlighted are the data



(a)



(b)



(c)

Figure 3.15: Position of EM tracker compared to Leap (Index MCP joint) (a) x axis; (b) y axis; (c) z axis.

points at which the tracking error was the largest. The mean 3D positional errors for the index tip, thumb tip and index MCP joint are 5.75 mm, 6.13 mm, and 4.81 mm respectively. However there are several spikes in error that are much larger than the mean, skewing it, where the maximum error was 26.79 mm, and 40.73 mm for the index tip and thumb tip respectively. The median errors were 4.5 mm, 5.22 mm, 4.3 mm. The index MCP joint was tracked with better accuracy, since it doesn't become occluded easily. Since the position of these joints control the orientation of the end effector, errors this large cause major errors in its orientation.

Knowing from the previous experiments that the roll angle of the hand influenced the tracking accuracy, the error was compared to the roll angle of the hand. The largest errors occurred when the roll angle was the largest in the negative direction, i.e., rotated so that the palm was facing to the right. Figure 3.17 shows the roll angle of the hand, indicating the data points where the error was the largest all occurred when the roll angle was near the minimum value recorded. The error is not only a function of the roll angle however. This was demonstrated in Figure 3.8, where the finger tracking performance depends on the x axis translation of the hand. Although the z axis does play a part in the accuracy of the sensing when moving farther from the sensor, this study assumed that the hand was within the acceptable tracking space in the z axis, so it was not compared. The 3D error plotted as a function of x–y position and roll angle can be seen in Figure 3.18, with roll angle as the vertical axis. The magnitude of the position error is represented on a blue to red color scale. This was used to visualize where the poor tracking occurred. The

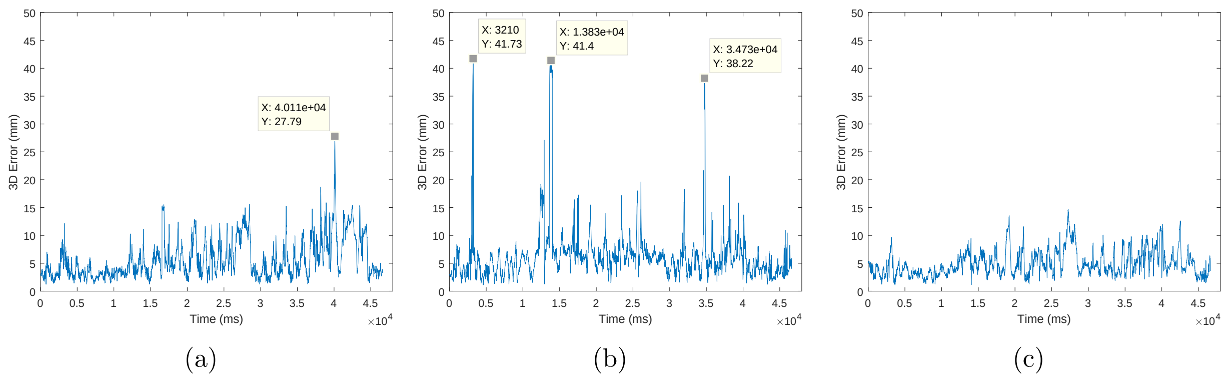


Figure 3.16: 3D position error of Leap sensor. **(a)** index tip; **(b)** thumb tip; **(c)** index MCP.

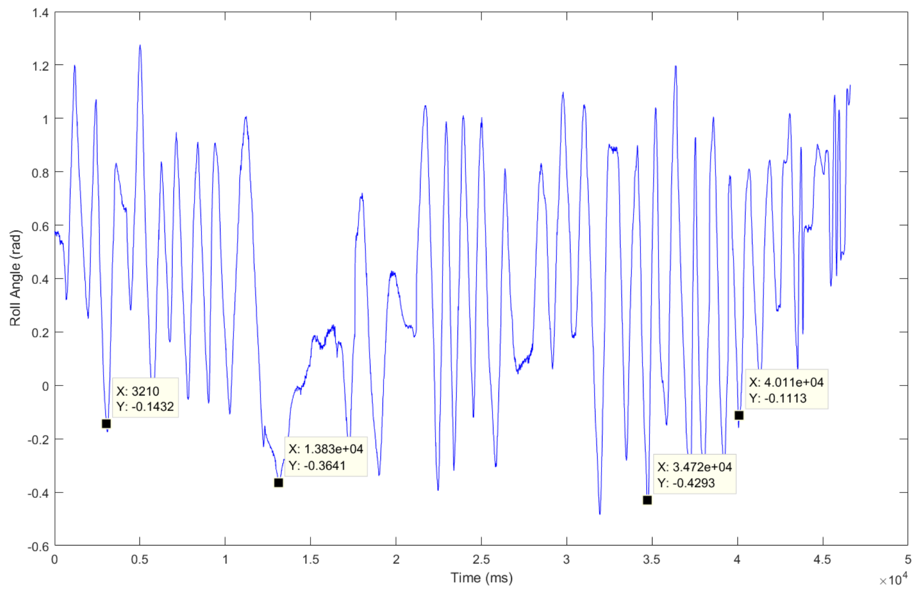
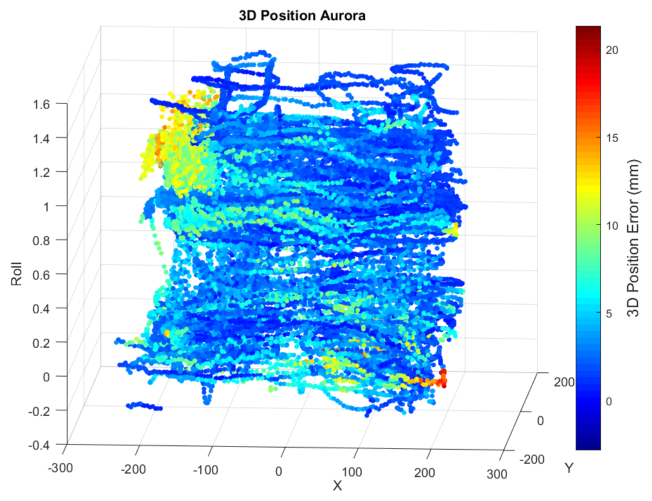


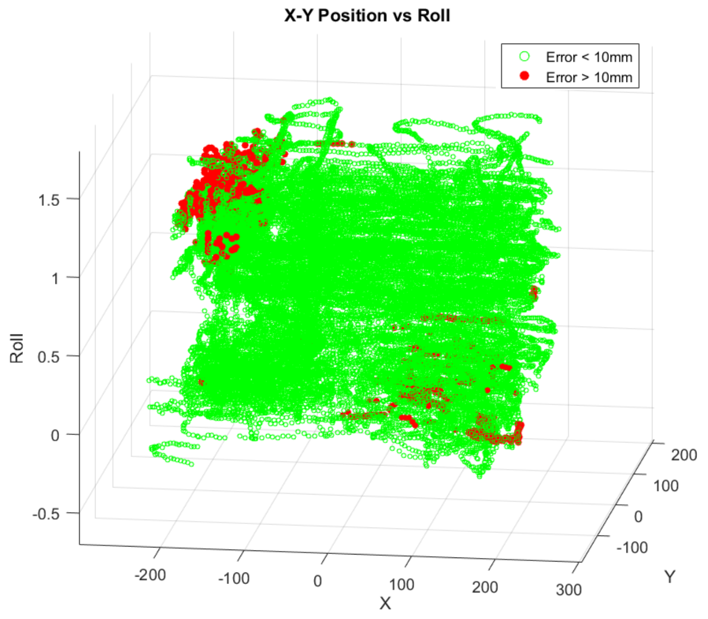
Figure 3.17: Roll angle of the hand

position error was deemed poor if the error was greater than 10 mm. The data extracted to show the points where the position error was greater than 10 mm are shown in Figure 3.18b, where red data points indicate poor tracking. Looking at these data, the x axis seemed to have the strongest correlation with error, while the y axis did not. A 2D plot of the roll vs. x axis position is shown in Figure 3.19. There are two regions in which the tracking becomes poor. These are when the roll angle of the hand is less than 0 rad, while in the negative x axis, and when the hand is rolled past 1 rad (approx. 60 degrees) in the positive x axis.

These data define where the useable workspace is for the sensor. For the right hand, the useable



(a) Error as a function of x and y position vs Roll angle



(b) Extracted data

Figure 3.18: Leap Motion controller position error 3D

workspace is defined as 250 mm in the x axis to the left of the Leap Motion controller for roll angles less than 0 degrees, and 250 mm to the right of the sensor for roll angles above 60 degrees. In between 0 and 60 degrees of roll, the trackable area is 250 mm to either side of the sensor. The size of the workspace for both controllers is approximately 120 mm in either direction in the y axis, and from 100 mm to 400 mm above the sensor in the z axis. This results in a trackable workspace of (250 mm $\times$ 240 mm $\times$ 300 mm) or 0.15 m^3

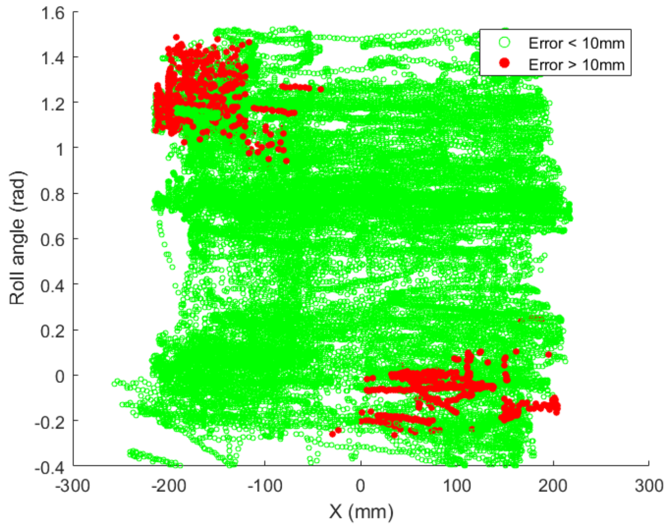


Figure 3.19: Error as a function of roll angle and x axis position

3.3.3 Final Set up

3.3.3.1 Sensor Placement

In order to compensate for the areas of poor tracking, two sensors were used to track one hand depending on the position and roll angle of the hand. Combining the sensors, we were able to track the hand for all roll angles with high accuracy. The two sensors were placed so one is viewing the hand from the left, and one viewing from the right. Depending on the roll angle of the hand, the input switches between the two controllers. Since the workspace of each hand is directly beside one another, one controller is placed in the middle, and used as the first controller for both the right and left hand. This middle controller is used for both hands when they roll inwards past 60 degrees, and the two side controllers used for when the hand is rotated outwards (counter-clockwise for the right, and clockwise for the left). This reduces the amount of controllers needed from 4 to 3, and still ensures that the hand is tracked for all roll angles that the hand is in during surgery.

The secondary sensors are positioned laterally, translated in the x axis by 320 mm and the y axis by 40 mm, and rotated inwards towards the centre by 15 degrees about the y axis. This places the center of the usable workspace for both sensors at the same point. The 15 degree angle allows an additional 15 degrees of tracking as the hand rolls farther counter-clockwise. Figure 3.20 shows the schematic of the final sensor placement, and the usable workspace for the right hand. The solid box indicates the workspace from the main sensor for roll angles greater than 60 degrees, and the dotted line the workspace for the side sensor for roll angles less than 0 degrees. The left hand is the mirror image of this, and Figure 3.21 shows this setup with both hands being tracked by two sensors simultaneously.

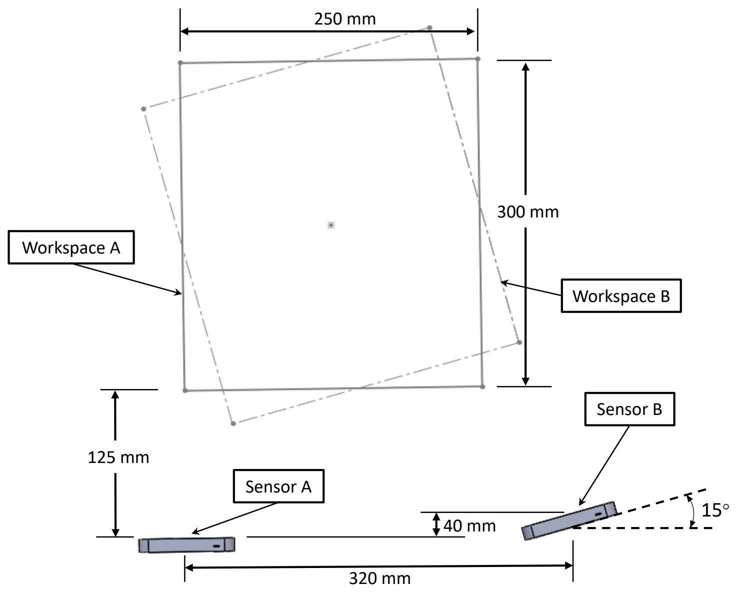


Figure 3.20: Controller placement and workspace

Mounts were 3D printed and mounted to a machined surface to align the controllers accurately as seen in Figure 3.22. One controller was connected to the Linux PC of the dVRK, one on the virtual machine of the Linux PC, and a third controller connected to a PC running Windows 10 using Transmission Control Protocol/ Internet Protocol (TCP/IP) to communicate with the ROS network.

Now that the controllers' final position and orientation are determined, the transformation from the secondary sensors to the middle sensor, from Equation 3.19, becomes constant using the position and orientation of the sensor relative to the middle sensor. For the right hand this

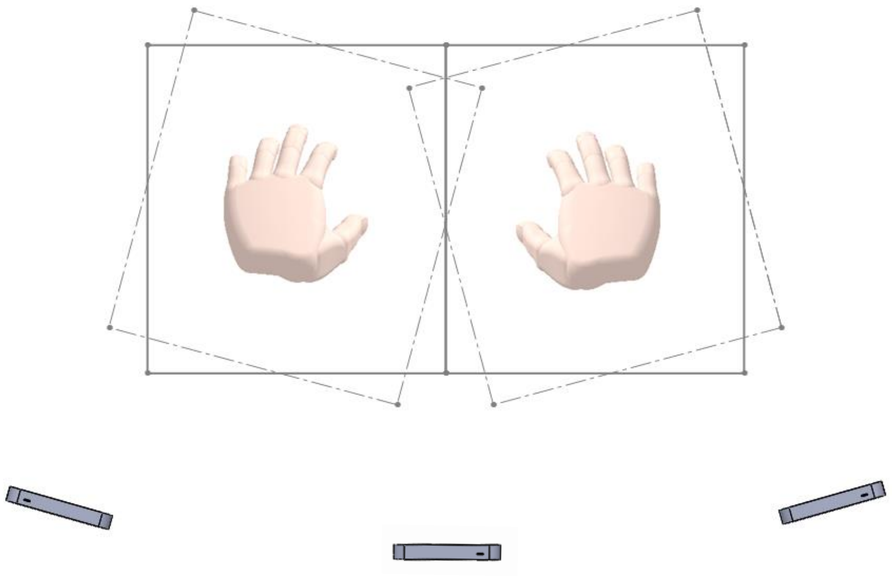


Figure 3.21: Full setup with multiple controllers

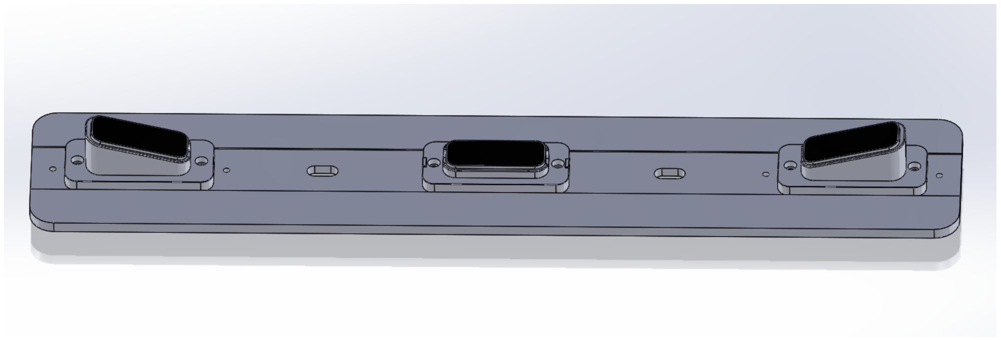


Figure 3.22: Mounting of Leap Motion Sensors

transformation is:

$$
{}^{B}\boldsymbol{T}_{S_R} = \begin{bmatrix} \cos(15^{\circ}) & 0 & \sin(15^{\circ}) & 320 \\ 0 & 1 & 0 & 0 \\ -\sin(15^{\circ}) & 0 & cos(15^{\circ}) & 40 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.25}
$$

And for the left hand, the transformation is:

$$^{B}\boldsymbol{T}_{S_L} = \begin{bmatrix} \cos(-15^{\circ}) & 0 & \sin(-15^{\circ}) & -320 \\ 0 & 1 & 0 & 0 \\ -\sin(-15^{\circ}) & 0 & cos(-15^{\circ}) & 40 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.26}$$

These matrices are used to fuse the data from the other sensors into the same coordinate system as the main sensor.

Now that the transformation is finalized, the conversion works much better than the fast calibration method in Section 3.3.1. Figure 3.23 shows the 3D error between the position reported by the two sensors for the index tip when the roll angle of the hand is between 0 and 60 degrees. This is important, because within this range of roll angles is where the two controllers will be switching between one another. Out of this range, when an occlusion occurs, the error between the two dramatically increases. The mean error is now 3.8 mm, with the maximum error being 8.5 mm. Since there is always a small error between the two sensors, there will be a slight jump in position when switching between the two. This is addressed in the following section.

3.3.3.2 Controller Weighting

Since there is still some error between the two sensors, there is a noticeable jump when controlling the end effector when switching between the two. In order to compensate for this, each data point that is used to control the pose of the PSM end effector (i.e., finger tips, MCP joint, and wrist) is computed as a weighted average between the two sensors. Let *weight1* and *weight2* be the weighting of the middle controller and the side controller respectively, then the weighted average of a joint position is:

$$\boldsymbol{p}_{avg} = \boldsymbol{p}_1 weight1 + \boldsymbol{p}_2 weight2 \tag{3.27}$$

The two weight values range between 0 and 1, and are computed based on the roll of the hand. Since there are values at which each sensor does not track well, these values are used as the thresholds for when the weighting causes tracking to be controlled by a single sensor. When the

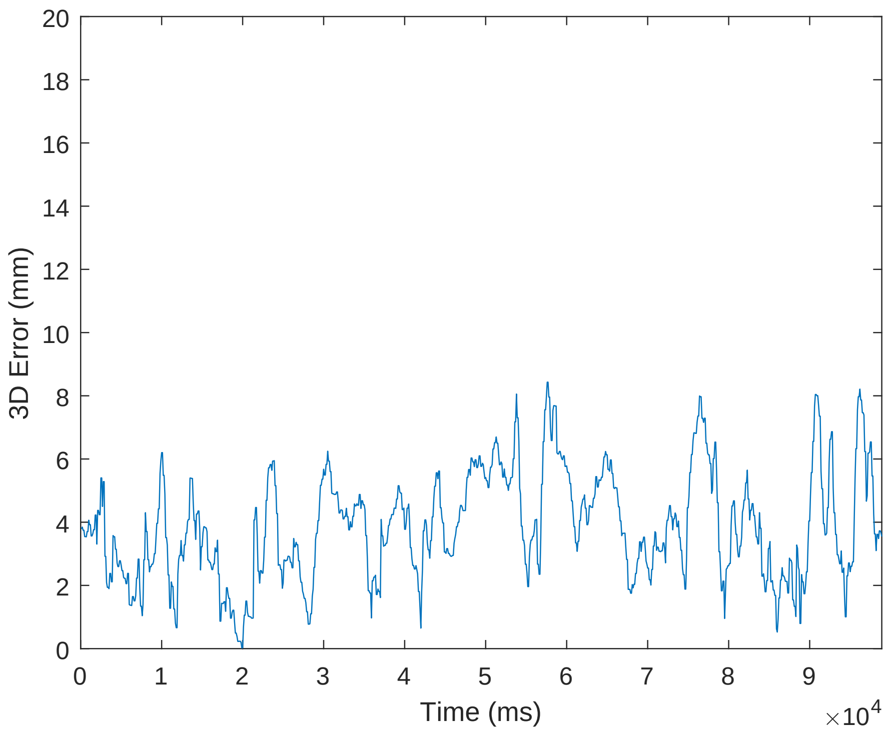


Figure 3.23: 3D position error with hard coded transformation

roll angle is greater than 60 degrees, weight1 = 1, and weight2 = 0. The opposite is true when the hand is rotated to less than 0 degrees: weight1 = 0, and weight2 = 1. In between these values, the weightings are a linear function of the roll angle. Figure 3.24 shows the weighting of the two controllers as a function of the roll angle. Highlighted is the area between 0 and 60 degrees where the roll angle is weighted using Equation 3.28. When the roll of the hand is in between the two threshold values of 0 and 60 degrees, weight1 is computed as:

$$weight1 = 1 - \frac{1}{60}roll \tag{3.28}$$

Then weight2 is computed as

$$weight2 = 1 - weight1 \tag{3.29}$$

such that the two controllers weights always add together to 1. Using this algorithm, when the hand is in a relatively neutral position, the hand is tracked by a combination of the two sensors.

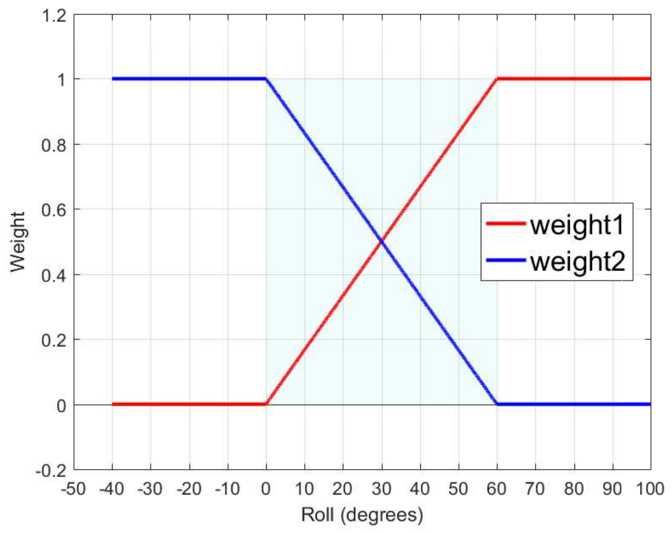


Figure 3.24: Multiple Leap Motion sensor weighting

As the hand rolls towards one of the extremes, the weighting of the controller increases until it reaches the threshold where the hand is then tracked only by the single sensor.

To visualize this method in use, the hand was placed in the workspace of the two sensors, and rotated slowly back and forth about the roll axis several times. The weighted average method was compared with simply switching between the two sensors when at a roll angle of 30 degrees (half way between the thresholds). Figure 3.25 shows a zoomed in plot of the position of the index finger from the two individual sensors, as well as the data when switching between the two controllers based on the roll angle of the hand. The black line indicates the basic switching, and the green line is the weighted average switch. The black line jumps instantly from one sensor to the other, whereas the green line has a smooth transition. This allows the switching between the controllers without a noticeable jump in the position control of the tool.

The EM tracking system was again used to compare the error, this time comparing the EM data with the weighted average data. The hand was again moved around the workspace in motions seen when operating, and the weighted average of the two sensors was used. The hand was rolled from -30 degrees to +90 degrees. Figure 3.26 shows an example of the error in the thumb tip position. The position from the EM tracker is in red, the position from the main sensor in blue, and the switching in green. As the hand is moved into an orientation that is not tracked well,

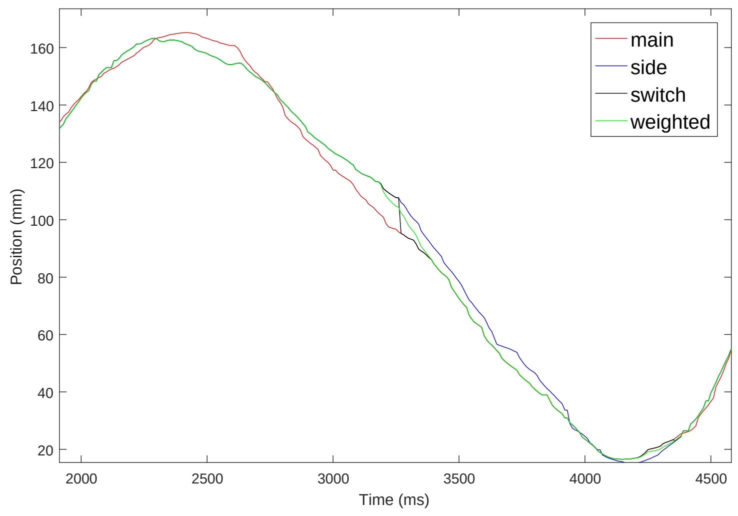


Figure 3.25: Switching algorithm visualized

the blue line (single sensor) deviates from the red line (EM) by as much as 50 mm, however, the green line follows the red very well. Figure 3.27 shows the error plotted vs. time for the joints

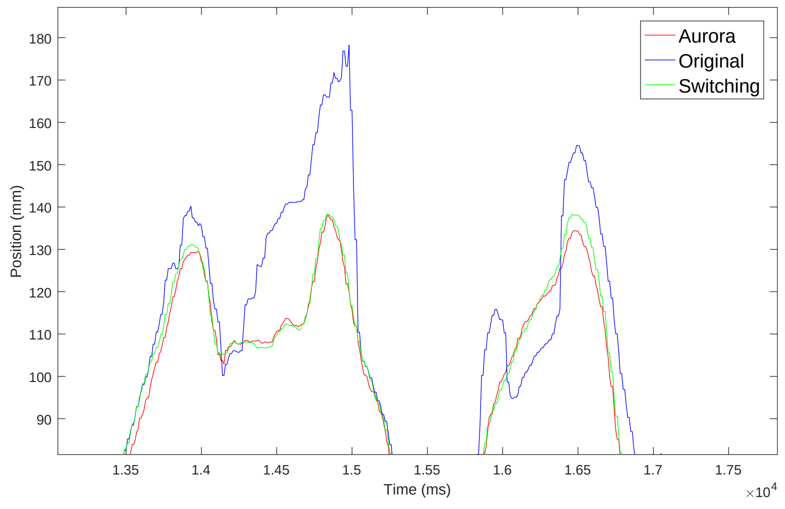


Figure 3.26: Position of thumb tip highlighting sensor switching.

of interest, comparing the data from the same trial. The error from the single sensor, as well as the error when using the weighted average is shown. Figures 3.27a–3.27c show the error using the algorithm, and Figures 3.27d–3.27f, the error using the single sensor. There is much improvement over the original with only one sensor when using both sensors. The mean error for the index tip, thumb tip, and index MCP joint were 2.9 mm, 3.1 mm, and 2.8 mm respectively. There are no longer any spikes in the error as there were previously, indicating that the hand was tracked well in all orientations the hand was placed in within the defined workspace.

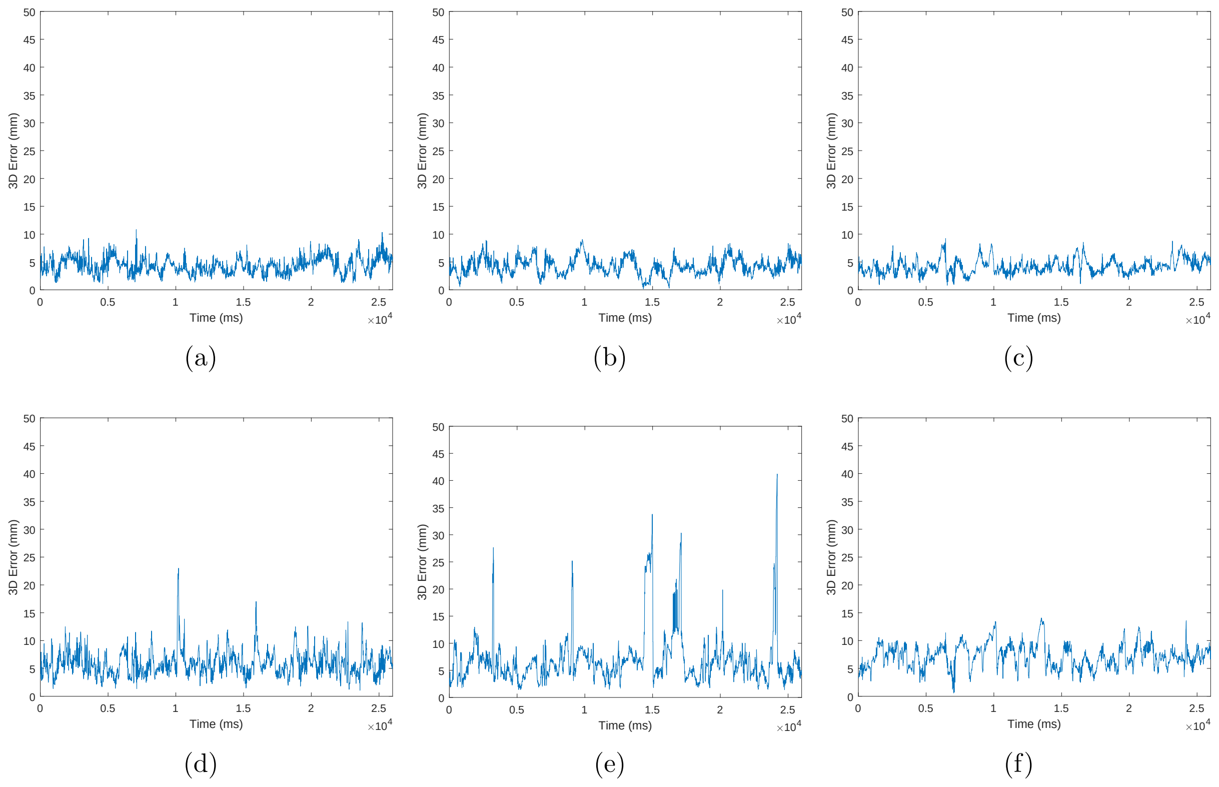


Figure 3.27: 3D position error of Leap sensor with final placement (a) index tip; (b) thumb tip; (c) index MCP, (e)-(f) show the same trial using the original sensor only

3.4 Conclusions

A novel method of interaction with the da Vinci robot has been developed, allowing the user to operate with natural hand motions. Using an inexpensive infrared tracking system, the hand position and orientation are mapped to control the da Vinci tools. An analysis of the workspace for a single sensor was provided, indicating that the tracking area is not suitable for the range of motion needed in surgery. This analysis provided the recommendation of the usable workspace, and the position of a secondary sensor in order to improve the robustness of the tracking. The result is a system that can track the hand and fingers with high accuracy in the range of motion needed for surgical tasks with a workspace of 0.15 m^3. Preliminary evaluation of the system is presented in Chapter 5.

Chapter 4

Endoscopic Camera Control

The previous chapter outlined the control of the PSM arms, and tools. This section outlines the work done on control of the endoscopic camera manipulator (ECM), and the video feed provided to the user.

4.1 Introduction

In traditional laparoscopic surgery, the surgeon relies on a camera assistant to position the camera for them. Cooperation and communication between the surgeon and the assistant is important for achieving optimum visualization. This results in increased distractions and surgery time, and can be difficult for the assistant to achieve the optimal view for the surgeon [93]. In RAMIS, including the da Vinci system, surgeons have the ability to control the camera themselves. Camera control is an important skill in RAMIS, as optimally viewing the surgical site can reduce unwanted errors. Maintaining the tools within the workspace is important in order to reduce accidental errors such as clipping of vessels [94]. The mimic dV-Trainer$^{\text{TM}}$ used for training surgeons, also emphasizes the control of the camera with several activities for positioning it [32]. This is because it is a difficult technique, where improvements can be made. To address this need, a novel method of endoscope control is proposed, using inertial sensors tracking the movement of the surgeon's head. This chapter outlines a novel method for endoscope control using a VR headset that tracks the users head movements. Also outlined is the method of replacing the stereo viewer in the da Vinci master console, with a head mounted display. Since the master console has been replaced in the previous section, the replacement for the viewing system is to place it directly on the surgeon, removing the need for the large console and monitors.

4.2 Literature Review

To improve upon the method of controlling the camera with an assistant, there has been work in creating robotic systems for positioning the camera, removing the need for the assistant. The Automated Endoscope System for Optical Positioning (AESOP), was developed by Computer Motion (Santa Barbara, Ca, USA), and allowed the surgeon control of the endoscope. The newest generation of the device is voice controlled [95]. The device was found to create a loss of comfort for the surgeon, and increase in operation time [96]. Another voice controlled interface is the ViKY EP (EndoControl, Grenoble, France), removing the need for an assistant, but has also shown to increase surgical procedure time [97]. Another commercially available system is the EndoAssist™ which allows the surgeon to move the camera with a foot pedal and head movements tracked by an infrared camera [98]. More recently, research has been done in creating autonomous camera systems [99–102], which work to center the tools in the field of view. These systems generally use image processing to identify the position of the tools relative to the camera, and then move the camera to keep the tools in view. This results in the movement of the camera without any explicit instruction or human direction. The disadvantage to these systems is that having the camera move at all times when moving the tools can be disorienting, with unwanted movements causing errors. Other research has used eye gaze tracking for endoscopic camera positioning [103], however the tracking technology remains relatively unreliable resulting in a different performace based on the environment and the user. [97].

4.2.1 da Vinci ECM

The endoscope of the da Vinci system is controlled by the endoscopic camera manipulator (ECM), which is a 4 DOF actuated manipulator, with joint sensors for control. Figure 4.1 highlights the axes of motion of this manipulator. The first two joints, θ_1, and θ_2, control the yaw and pitch of the endoscope about the remote center of the shaft. These two joints are the main positioning joints which physically move the entire orientation of the arm. The third axis is the insertion axis of the endoscope along the axis of the shaft, d_3. It is used to zoom in and out, and does not change the center of the viewing area. The last axis, θ_4, provides the roll of the camera about the endoscope shaft, which adjusts the horizontal axis in the viewing screen. To reposition the endoscope, the yaw and pitch joints pan the field of view, and the fourth joint, roll, adjusts the orientation. The first two joints are the most important, and move the view of the endoscope, while the roll joint adjusts the orientation after the desired target is in view.

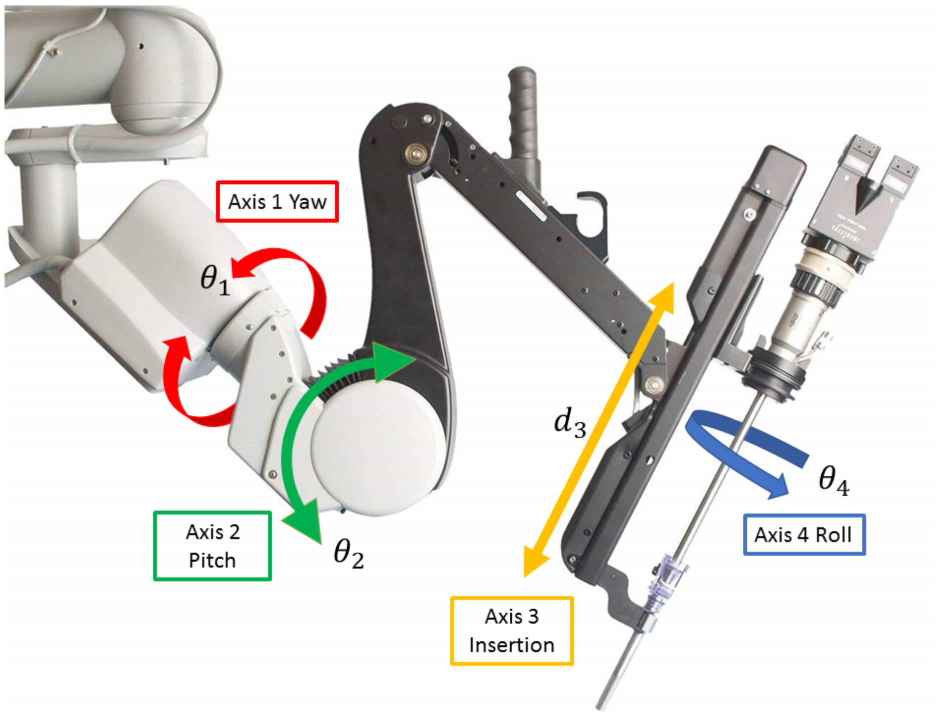


Figure 4.1: Joints of the ECM and their direction of motion

The current method of controlling the camera position with the da Vinci is to use an additional pedal for switching control to the ECM arm. When the camera pedal is pressed, the master manipulators are disengaged from the PSM arms, and the ECM becomes the controlled slave. When the pedal is released, the ECM remains in position, and the system resumes normal operation. The two MTMs work together to control the camera positioning. The surgeon can zoom in and out, and pan and tilt the viewing field to visualize the structures of interest. This method of control takes some time getting used to, and is not the most intuitive. It also requires the surgeon to stop what they are doing, and move the camera before returning to operation. A different method of controlling the endoscope is proposed here, using the natural motions of the head to look around. When you want to move the camera to view a target out of view, simply rotate your head as you would naturally. The proposed solution is to track the movements of the head using inertial sensors to control the the joints of the ECM arm. This method may be more intuitive for users, as one will be able to just rotate their head as they would naturally, to reposition the view from the endoscope.

4.2.2 RViz Simulation

To test and validate the control of the ECM, a simulator was used, developed for the dVRK, and built on RViz [84]. RViz is a 3D visualization tool for ROS, allowing researchers to implement robotic control without the necessary hardware. Figure 4.2 shows the simulator, with the ECM in the middle between the two PSM arms. The arms can be moved in the same manner as with the real da Vinci robot, through the same ROS topics provided with the dVRK. In order to visualize the camera movements, this simulator was modified by adding a virtual camera at the tool tip of the ECM to view the movements of the tools as the camera is rotated. This camera is placed with its frame of reference such that it is fixed to the end of the endoscope. Figure 4.3 shows the simulator with the robotic arms, and the virtual camera window for visualizing movements.

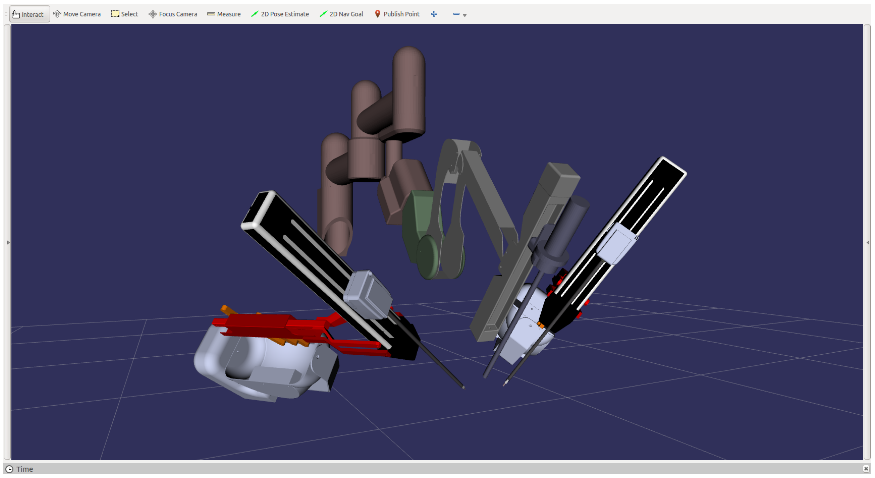
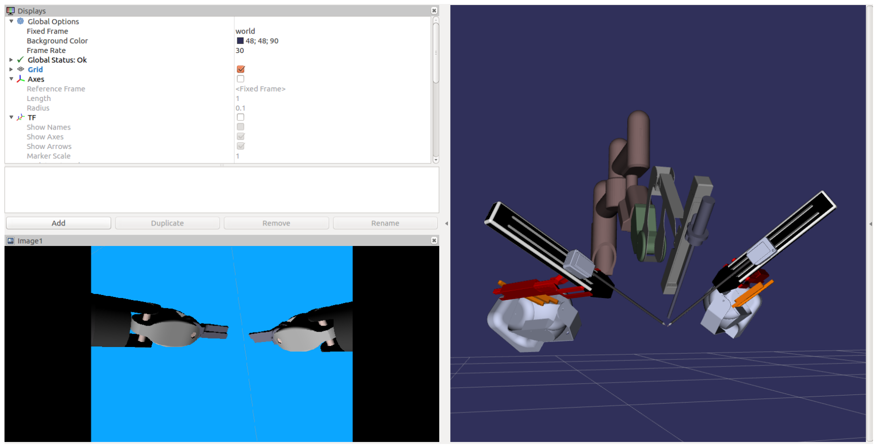


Figure 4.2: RViz based simulator for the da Vinci robot



```
Global Options
  Fixed Frame          world
  Background Color     48; 48; 90
  Frame Rate           30
Global Status: Ok
Grid                   ✓
Axes                   ☐
  Reference Frame      <Fixed Frame>
  Length               1
  Radius               0.1
TF                     ☐
  Show Names           ☐
  Show Axes            ✓
  Show Arrows          ✓
  Marker Scale         1
```

Figure 4.3: Modified simulator showing robotic arms and endoscopic view

4.3 Endoscopic Camera Control

4.3.1 Initial Prototype

Before the headmounted display was implemented, an inertial measurement unit (IMU) was used to assess the feasibility of controlling the camera arm with head movements.

In order to control the camera arm with the head, the orientation of the head is mapped to the ECM. The head can be assumed to be a rigid body, and therefore the orientation of the head can be characterized by the three Euler, or RPY angles as seen in Figure 4.4 [104]. Where roll is tilting about the axis from the back to the front of the head (side to side bending), pitch is a rotation about the axis from ear to ear (neck flexion and extension), and yaw a rotation about the axis from the front to the back of the head (lateral rotation; i.e., looking side to side). The endoscope also has three DOF in rotation, therefore these three rotations of the head can be mapped to the rotations of the ECM.

To measure the orientation of the head, an IMU was used (sparkfun 9 DOF IMU: LSM9DSL). The 9 DOF comes from the device housing three separate sensors: a tri-axis accelerometer, a tri-axis gyroscope, and a tri-axis magnetometer, measuring linear acceleration, angular velocity, and magnetic field respectively. The sensor was placed in a 3D printed case, and attached to a head strap, see Figure 4.5. The sensor was placed on the back of the head, capable of measuring

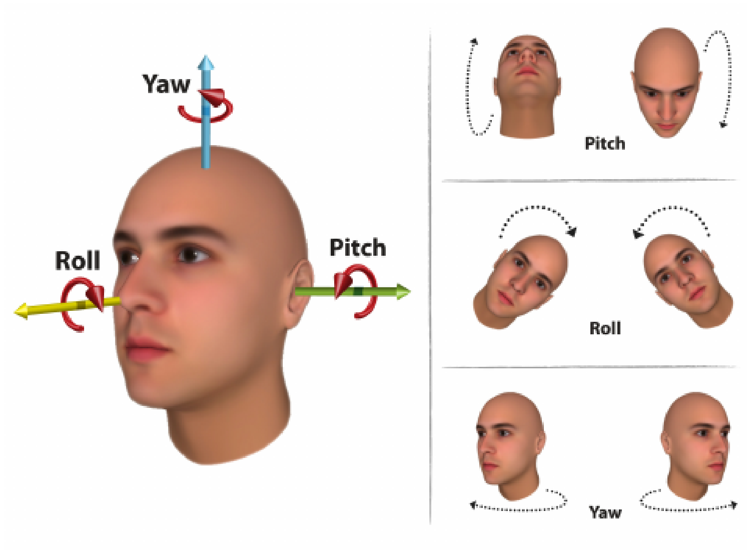
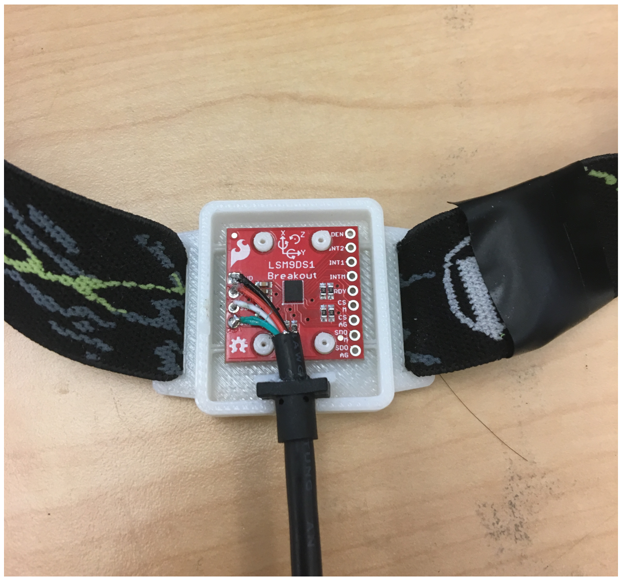
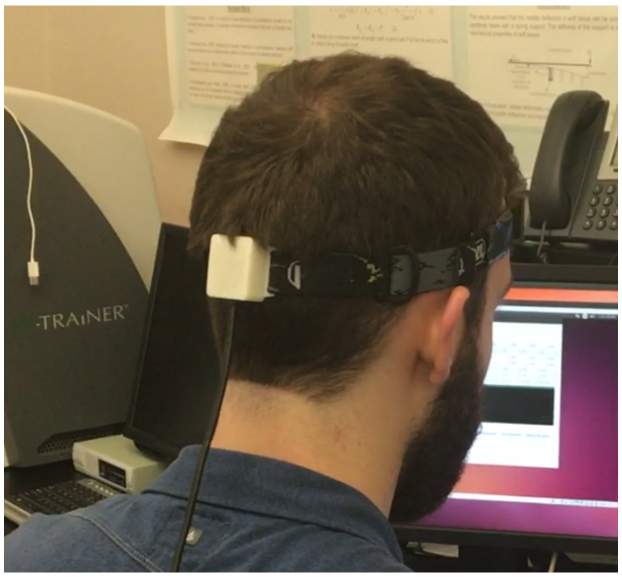


Figure 4.4: The axes of rotation of the head in terms of pitch, roll, and yaw angles © 2015 IOS Press [104]

the rotations of the head about the three axes. The IMU was interfaced with Arduino for ease of implementation.



(a) IMU board in 3D printed case



(b) Strapped to head aligned with axes of rotation

Figure 4.5: Head mounted IMU

4.3.2 Method of Control

Communication with the ECM occurs in the same manner as with the PSMs, through the dVRK-ROS interface. To integrate the IMU with the ROS network, allowing communication between the arduino and the dVRK, the ROS library *rosserial_arduino* was used. This effectively turns the Arduino board into a full ROS node, that polls the IMU and publishes the data through the serial communication library. The signals from the IMU are polled at 100 Hz, and published through ROS, and can be subscribed to by other nodes on the network. An additional node was created that takes these data, computes the desired ECM joint angles, and publishes it to the dVRK. Unlike in Chapter 3, where the PSM was controlled in Cartesian space, the ECM is controlled in joint space. This is because the 3 DOFs of the head match up with the 3 rotational DOF of the ECM. A rotation of the head about a specific axis sends a command to rotate the ECM about that same axis, (i.e., when you look to the left, this is a rotation about the z axis, or yaw, and this will cause the first joint of the ECM to rotate). There is no translation of the head that is used, so it made more sense to control in joint space, and map one rotation to another. The ROS topic provided for controlling the ECM joints is `/dvrk/ECM/set_position_joint`. This topic takes a set of 4 joint variables corresponding to the 4 joints of the manipulator. Each joint can be controlled individually with commands to the desired joint angle.

It was found that controlling the roll angle of the tool using the roll of the head is very uncomfortable. When looking around naturally, one does not use the roll angle of the head very often (i.e., tilting head to side), as the horizon line tends to be kept fairly horizontal. For this reason, only the yaw and pitch angles of the head are used to control the position of the ECM. The other two DOFs of the ECM are then controlled using the hands, to zoom and tilt the view of the screen.

Two methods were proposed for controlling the endoscope during surgery with head tracking:

1. *Relative motion with pedal*: This method works in the same manner as the da Vinci currently does. The camera is not controlled until the user presses the camera pedal. This allows the user to freely reposition their head during operation, when the pedal is not pressed.

2. *Always controlled*: control of the camera is done with the global orientation of the head.

This method will allow the user to look around while not having to interrupt the control of the tools. However, it will not respond to movements until the angular velocity of the head is above a threshold. This ensures that the user can operate while their head is still, then move the camera when they desire, filtering out unwanted movements.

Method 1 works in a simple manner, by using the camera pedal to clutch between control of the tools and the ECM arm. Since the orientation of the head is only taking control during the pedal press, it is not crucial to have an absolute orientation of the head with respect to a world frame of reference. Instead, the relative orientation of the head is computed during the time the pedal is pressed. Since the movement of the head only occurs for a short period when the pedal is pressed, drift is not as big a concern, and the angular position of the head can be computed by integration of the angular velocity from the gyroscope. The gyroscope is polled by the Arduino at 100 Hz, and then publishes the angular velocity of the head in radians as a vector $\boldsymbol{\omega_h} = \left[\omega_{h_x}\ \omega_{h_y}\ \omega_{h_z}\right]^T$ to the ROS network at this same rate. The angular position, $\boldsymbol{\theta_h} = \left[\theta_{h_x}\ \theta_{h_y}\ \theta_{h_z}\right]^T$ can be computed by the integral of the angular velocity over time:

$$\theta(t) = \int_0^t \omega(t)dt \tag{4.1}$$

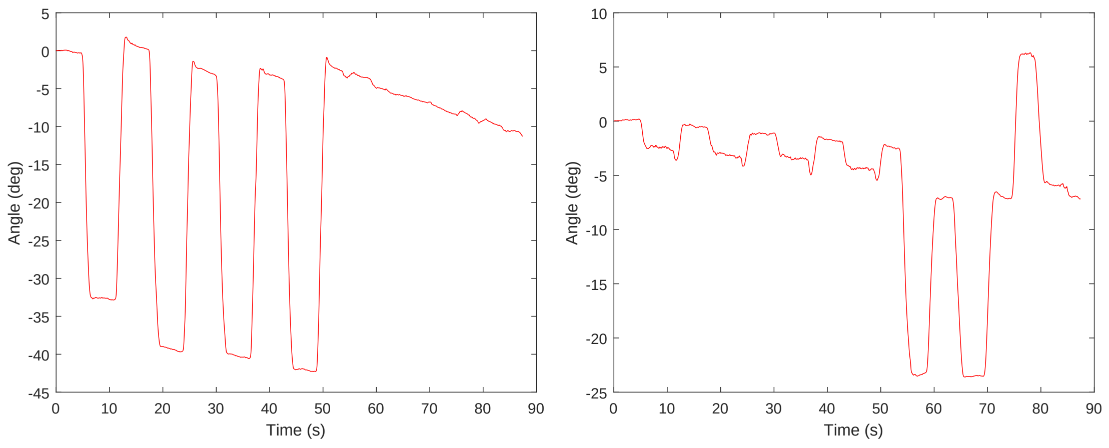
Where time t=0 is the moment the pedal is pressed. The approximation for a digital system is a sum of finite numbers taken at the sampling period T_s, so this becomes:

$$\theta(t) \approx \sum_0^t \omega(t)T_s \tag{4.2}$$

Where $T_s = 0.01\,\mathrm{s}$. This approximation of angular position with a gyroscope will result in drift over time, due to small errors accumulating due to the finite sum. As mentioned, it is not of concern when moving relative to an initial position. If the gyroscope was to be used as the orientation over a long period, the orientation would eventually drift, causing the user to have to rotate their head to achieve the same original position. When the pedal is released, the angular position is reset to zero for the next time it is pressed.

The main issue with this method is that this in not much of an improvement over the current clutch system with the da Vinci. It may be more intuitive moving the head instead of moving with the master controls, but the surgeon has to clutch in the same manner, and will still have to interrupt the control of the tools in order to reposition the camera. For this reason, the second method was proposed, where the surgeon will be able to pan the camera at will with their head while continuing to control the tools.

Method 2: In this method, the orientation of the head controls the camera at all times. Therefore the orientation of the head with respect to a fixed reference frame must be known. This is because in this case drift will be a concern, as the orientation is always being controlled when operating. Since yaw of the head is to be measured, the accelerometer cannot be used for this axis, as the gravitational vector is parallel to it. Attempting to use the gyrosope data only for the angular position of the head results in drift over time, resulting in a different measured position when returning your head back to the initial starting point. Figure 4.6 shows this drift as the head was moved in a simple task looking between two points, resulting in mainly yaw rotation, then looking between points resulting in pitch rotation. When the sensor was returned to the same initial position, the measured angle had drifted substantially. After 90 seconds, the drift was measured to be -10 degrees in yaw and -7 degrees in pitch.

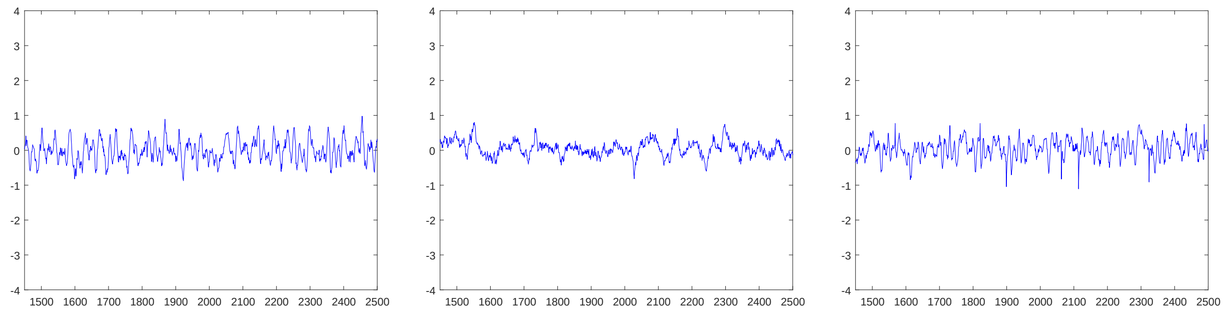


(a) Yaw (b) Pitch

Figure 4.6: Angular position using gyroscope showing drift

There are many different methods of using a combination of the sensors in the IMU to get the orientation of a rigid body, including: (i) A gradient descent algorithm from [105], and (ii) a Kalman filter for quaternion based orientation [106], which both filter and fuse the data from gyroscope, magnetometer and accelerometer. It was decided to use the method in (i), as open source code was available to implement easily. It provided a stable orientation without any drift, due to the magnetic field reference of the magnetometer.

When wearing the sensor on the head there was still some noise in the orientation, from sensor noise, as well as involuntary motion tremor. When operating, the camera should remain still when attempting to hold the head still. This tremor and noise caused unwanted motion of the camera. Filtering the data further is a possibility, but in order to not introduce delays, a different method was used. The angular velocity of the head when under a certain threshold is used to prevent control of the ECM when keeping the head as still as one can. In order to determine the threshold to use for filtering out the small movements of the head, the IMU was placed on the head and then the head held still while performing a simple task using the da Vinci. Figure 4.7 shows the data from the IMU, showing the angular velocity of the head, while attempting to hold still after moving to look at a target. The average value for three different subjects was 1.2 degrees/sec. The threshold value was set to be 1.5 degrees/sec, just above this value. As the head is voluntarily moved, the angular velocity increases much higher than this, allowing the camera to move.



(a) (b) (c)

Figure 4.7: Angular velocity of head held still (a) x axis; (b) y axis (c) z axis

4.3.3 Testing

To test the feasibility of using head tracking to control the endoscope, a simple trial was run, rotating the head back and forth looking at three targets on a wall (Figure 4.8). How accurately

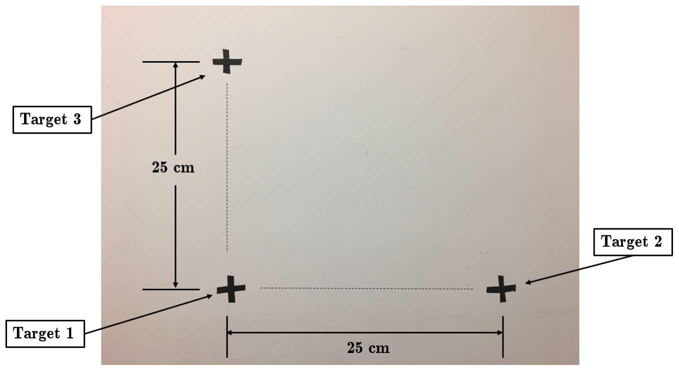
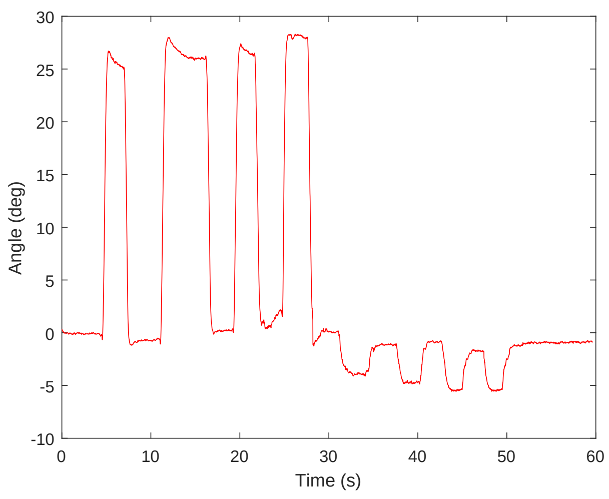


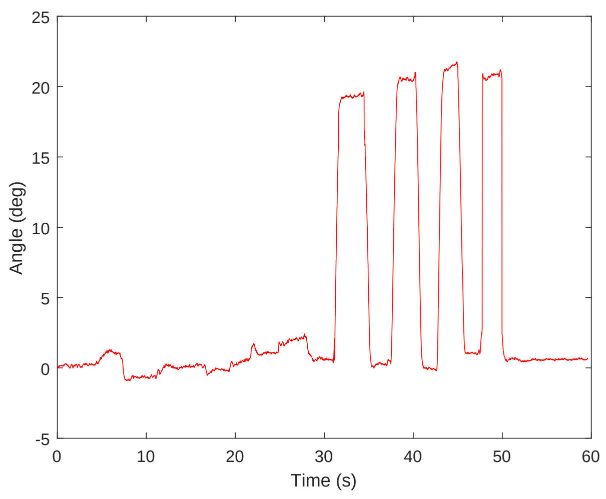
Figure 4.8: Targets used for head tracking evaluation

the user could maintain a position while looking at the target was measured. Target 1 was the starting position, and the other two targets were positioned so the user had to rotate their head in the yaw and pitch axes to look at them. The user was sitting two feet away from the wall, making looking to target 2 and 3 approximately a 25 degree head turn. The angles are reported as the angle relative to the initial starting position when looking at target 1. The trial was to move the head to look at Target 2 and hold for 5 seconds, then rotate back to Target 1 and hold for 5 seconds, repeating this 4 times. Then the same procedure was completed between Target 1 and Target 3 immediately after. The head was then rotated back to the starting position and held. Figure 4.9 shows the filtered orientation of the head during the tracking task, and the average orientation of the head for each target is shown in Table 4.1.

These data show how accurate the subject was able to move their head to the desired points. At the end, returning to the starting point, there was no drift, and the difference in angles was less than 1 degree. This shows that positioning the head and holding the head still looking at an object can be done with a fairly high precision. The variation in measured angle when attempting to return to the same position was ±1.5 degrees. Positioning the camera arm with your head is a promising solution, and further evaluation will be performed.



(a) Yaw angle



(b) Pitch angle

Figure 4.9: Head tracking evaluation data

Table 4.1: Head positioning task

	Target 1	**Target 2**	**Target 3**
Mean Position (deg)	0.26 ± 1.70	26.74 ± 1.58	20.33 ± 1.33

4.3.4 Yaw and Pitch Joints

When the pedal is pressed to begin control of the ECM, (camera pedal in Method 1, and safety pedal in Method 2) the current joint positions of the ECM are saved: $\theta 1_{initial}$, and $\theta 2_{initial}$ corresponding to Joint 1 and Joint 2 of the ECM respectively. The angular position of the head, θ_h is computed relative to the initial orientation when the pedal is pressed. This allows the user to position their head in a comfortable starting position. This angular position is used to send a command to move the ECM relative to the saved joint position. Letting θ_1, θ_2 be the yaw and pitch angles respectively, with subscripts corresponding to the joints of the ECM arm, the desired joint angles are computed as follows:

$$\begin{aligned} \theta_1 &= \theta 1_{initial} + \theta_{h_{yaw}} \\ \theta_2 &= \theta 2_{initial} + \theta_{h_{pitch}} \end{aligned} \tag{4.3}$$

However, these equations only work when the angle of the last joint, $\theta_4 = 0$. As a rotation is introduced in the last joint of the ECM, the view that the surgeon is placed in is not in line with the original frame of reference any more. As the head is rotated to pan the view horizontally, Joint 1 will still move in the same manner as previously. This results in panning the view of the screen on a diagonal. When there is roll introduced in the shaft of the endoscope, (i.e., $\theta_4 \neq 0$), the frame of reference of the surgeon has changed, and the movements need to pan according to this new frame of reference. Equations 4.3 becomes a function of θ_4 as well:

$$\begin{aligned} \theta_1 &= \theta_{1_{initial}} + (\theta_{h_{yaw}} cos(\theta_4) - \theta_{h_{pitch}} sin(\theta_4)) \\ \theta_2 &= \theta_{2_{initial}} + (\theta_{h_{pitch}} cos(\theta_4) + \theta_{h_{yaw}} sin(\theta_4)) \end{aligned} \tag{4.4}$$

The control of each of the first two joints becomes a component of the the pitch and yaw of the head when there is roll introduced into the shaft.

Both methods were briefly tested for ease of use, and both had their advantages. In method 1, the control does not start until the camera pedal is pressed, meaning the head can be freely moved without taking control of the endoscope. However, clutching whenever there is a need to move the camera results in disengaging the PSM arms, and a delay in operation. Method 2 allows one to look around at the same time the tools are operational. This method of control worked fairly well, and the camera arm did not respond to movements until the user deliberately moved. However, using the global position of the head to control the camera resulted in holding the head at an angle for an extended period to maintain the camera in a new position. To solve this issue, a pedal function was added that would disengage control of the system, allowing one to move the head back into a neutral position. When the pedal is released, the control of the ECM starts again, saving a new initial head orientation from which to move. So while a pedal is still used in this method, it involves much less clutching time than in Method 1. An additional function added is the ability to pause the control of the camera with a double tap on the pedal. This allows the surgeon to lock the position of the camera in place if planning on keeping a specific view for a long time. The two methods of controlling the panning of the camera will be tested against one another for user preference and performance.

4.3.5 Insertion and Roll Joints

The control of the last two DOF is done with the positions of the hands from the Leap Motion controllers when the camera pedal is pressed. This includes the roll of the endoscope shaft, and the insertion along the length of the shaft, controlling the magnification of the image. The roll of the camera is controlled by the angle that the vector between the hands makes with the x axis, as shown in Figure 4.10. The position of the left and right hand are $\boldsymbol{p_{h_l}}$, and $\boldsymbol{p_{h_r}}$, respectively, and

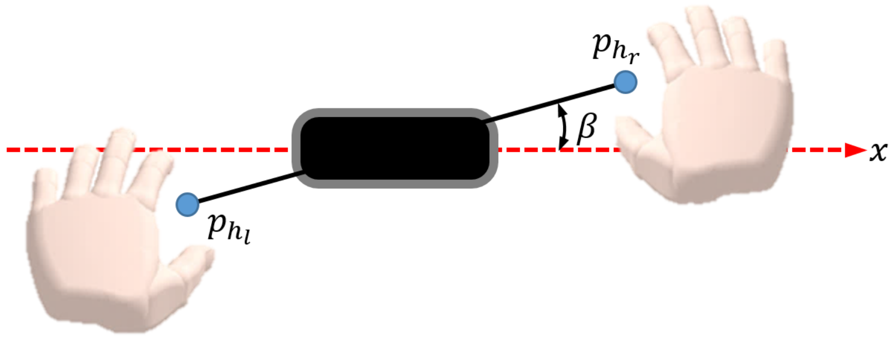


Figure 4.10: Endoscope roll angle from hands

the vector between the two hands is $\boldsymbol{v_h} = \begin{bmatrix} v_{h_x} & v_{h_y} & v_{h_z} \end{bmatrix}^T$. The angle between this vector and the x axis of the Leap Motion controller is:

$$\beta = \arctan(\frac{v_{h_y}}{v_{h_x}}) \tag{4.5}$$

Then, just as with Joint 1 and 2, the angle command sent to the ECM is a combination of an initial saved position and the relative motion after the pedal press. Joint 4 is controlled as follows:

$$\theta_4 = \theta_{4_{initial}} + \beta \tag{4.6}$$

The insertion of the endoscope shaft is simply computed by the relative distance the hands move from the origin of the Leap sensor, as shown in Figure 4.11. As the user brings their hands in towards their head, the endoscope will zoom in, as if pulling the view in towards their face. Bringing the hands towards the face, and the distance from the sensor increases in the y and z axes.

The midpoint between the two hands: $\boldsymbol{p}_{mid} = \frac{\boldsymbol{p}_{h_l}+\boldsymbol{p}_{h_r}}{2}$ is computed, then the relative movement of this point in the y–z plane is used to compute the translation of the third joint. The relative

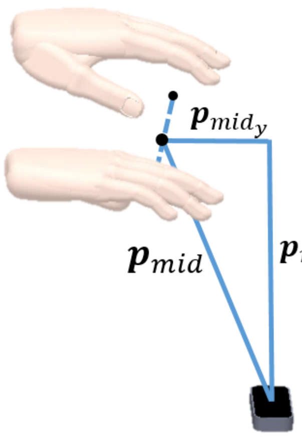


Figure 4.11: Endoscope magnification controlled by hands

distance travelled is computed as follows:

$$\Delta\boldsymbol{p}_{mid} = \sqrt{(\Delta\boldsymbol{p}_{mid_y})^2 + (\Delta\boldsymbol{p}_{mid_z})^2} \tag{4.7}$$

Then the desired command for Joint 3, (d_3) is computed in the following manner:

$$d_3 = d_{3_{initial}} + \Delta\boldsymbol{p}_{mid} \tag{4.8}$$

The translation of the third joint is measured from home position when the endoscope is fully retracted. Positive values increase as the endoscope is inserted towards the floor. So as the hands are moved in towards the head, the endoscope is inserted further, magnifying the image.

4.3.6 Motion Scaling

In order for the camera movement to look natural, the amount the camera rotates needs to match up with the amount the head rotates. This would be a 1:1 ratio if the center of rotation of the endoscope was right at the lens. This is not the case during a procedure. As the translation of joint 3, d_3 is increased, the distance to the target is smaller, however, the origin of rotation of the tool remains the same. The rotation occurs about the remote center of motion (RCM) of the ECM, constrained by the kinematics of the manipulator. When the endoscope is fully retracted, the lens is placed at the RCM, but during operation, the camera is inserted into the patient a distance d_3. When the image is zoomed in, the frame of reference of the surgeon is now placed closer to the tissue. Rotation of the endoscope the same angle as the head results in a distance travelled on the screen that is more than expected, and can result in an unnatural feeling, difficulties in positioning, and possible motion sickness. In Figure 4.12, the camera is placed a distance of y_1

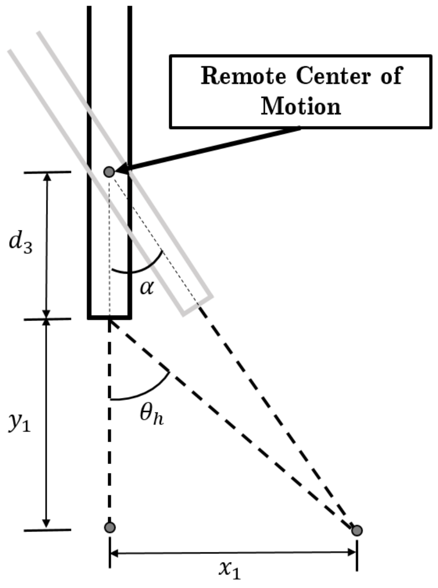


Figure 4.12: ECM scaling

from the tissue of interest. Then when the head is rotated an angle of θ_h, the rotation should result in a translation of the center of view of:

$$x_1 = y_1 \tan(\theta_h) \tag{4.9}$$

However, the control of the ECM does not rotate the camera at the lens position. This is because the RCM of the tool remains stationary, mechanically constrained by the manipulator, even while the camera lens is moved closer to the target. In order to have the view for the user translate the correct amount, the joint at the RCM needs to rotate to get the same amount of translation as if the centre of rotation was at the tip of the endoscope. The distance x_1 can also be put in terms of the desired joint angle α:

$$x_1 = (y_1 + d_3)\tan(\alpha) \tag{4.10}$$

Equating these two equations and solving for α, the desired joint angle to send to the ECM now becomes:

$$\alpha = \arctan\left(\frac{y_1}{y_1 + d_3}\tan(\theta_h)\right) \tag{4.11}$$

Where d_3 is the amount the tool has inserted past the center of rotation. So given the relative movement of the head $\theta_{h_{yaw}}$, and $\theta_{h_{pitch}}$, substituting in the angles of the head and joint angles of the ECM for α, the corresponding amount the joints move by is calculated as:

$$\begin{aligned} \theta_1 &= \arctan\left(\frac{y_1}{y_1 + d_3}\tan(\theta_{h_{yaw}})\right) \\ \theta_2 &= \arctan\left(\frac{y_1}{y_1 + d_3}\tan(\theta_{h_{pitch}})\right) \end{aligned} \tag{4.12}$$

This results in the rotation of the joints being smaller than the rotation of the head, scaled as the tool is inserted further. Since the tools are set up initially an arbitrary distance away from the patient, the amount the endoscope has travelled cannot be used to compute the distance d_1 from the tissue. Instead the distance between the endoscope tip and the tool tips, along the z axis of the endoscope is used to estimate the distance. This is a fairly good indicator of the distance between the camera lens and the target, since the tools are interacting with the tissue of interest.

4.3.7 Tool Rotation

When the camera is moved, this introduces a rotation of the ECM with respect to the PSM base frame. When the viewpoint from the endoscope is changed, the tool movements have to be remapped to this new frame of reference. When the ECM is in home position, the movement of the hands results in the tools moving accordingly, based on the rotation of the passive set up joints highlighted in Chapter 3. However, when the actuated joints provide an additional rotation, the movement no longer will match with the users hands when viewed through the endoscope.

To visualize the tool movements when the camera has been moved, rotations of the camera in the pitch, yaw, and roll angles were introduced and shown in the simulator. Moving the tool along the same path results in the tool moving incorrectly in the ECM frame of reference, as shown in Figure 4.13. When the ECM is in home position, a movement in the x axis results in the tool moving on the screen horizontally, matching the hand. In Figure 4.13b, a rotation of the ECM has been introduced, and the axes are no longer aligned. A movement of the tool in the same path as before now moves on the screen on a diagonal. The same thing happens with the other axes, where movements in the y and z axes no longer move accordingly, depending on the ECM rotation.

An additional rotation matrix is needed from the PSM to the ECM when the orientation of the endoscope has changed. This rotation matrix is computed as the combination of the rotations

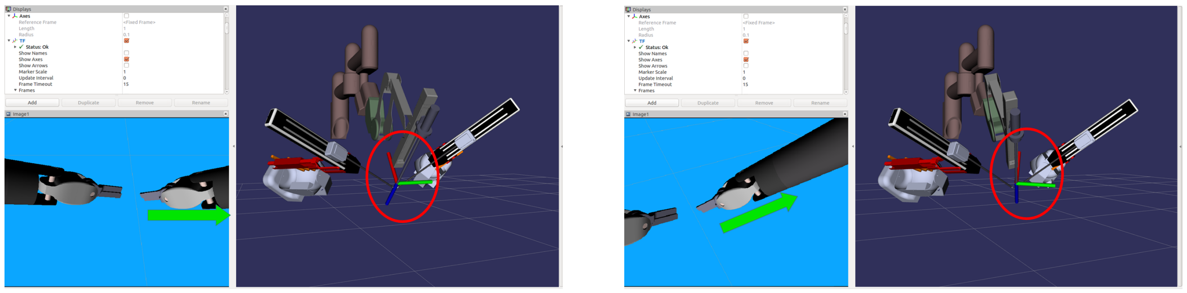

(a) Home position (b) Rotated position

Figure 4.13: View of endoscope visualized as rotation is introduced (a) ECM in home position; (b) ECM rotated

of the 3 rotational joints, working backwards from the camera frame of reference to the base of the ECM. Rotation matrices are created from the angles of the joints 4, 2, and 1:

$$\boldsymbol{R}_z(\theta_4) = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 \\ \sin\theta_4 & \cos\theta_4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

$$\boldsymbol{R}_x(\theta_2) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_2 & -\sin\theta_2 \\ 0 & \sin\theta_2 & \cos\theta_2 \end{bmatrix} \tag{4.14}$$

$$\boldsymbol{R}_y(\theta_1) = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 \\ 0 & 1 & 0 \\ -\sin\theta_1 & 0 & \cos\theta_1 \end{bmatrix} \tag{4.15}$$

These are multiplied to obtain:

$$\boldsymbol{R}_{ECM_A} = \begin{bmatrix} \cos(\theta_1)\cos(\theta_4) - \sin(\theta_1)\sin(\theta_2)\sin(\theta_4) & -\cos(\theta_2)\sin(\theta_4) & \sin(\theta_1)\cos(\theta_4) + \cos(\theta_1)\sin(\theta_2)\sin(\theta_4) \\ \cos(\theta_1)\sin(\theta_4) + \sin(\theta_1)\sin(\theta_2)\cos(\theta_4) & \cos(\theta_2)\cos(\theta_4) & \sin(\theta_1)\sin(\theta_4) - \cos(\theta_1)\sin(\theta_2)\cos(\theta_4) \\ -\sin(\theta_1)\cos(\theta_2) & \sin(\theta_2) & \cos(\theta_1)\cos(\theta_2) \end{bmatrix} \tag{4.16}$$

As the ECM is rotated, this rotation matrix is updated in real time. It is then premultiplied to Equations 3.14 and 3.16 to move the tool in the frame of reference of the endoscope. This ensures that the motion of the hands results in the same motion of the tools when viewed by the surgeon.

The head tracking developed up to this point used an IMU, with a magnetometer for a steady reference direction. Since this device is to be used in the operating room (OR), magnetic interference poses a problem with tracking. A different solution is presented to address this issue.

4.3.8 Oculus Rift Tracking

In the OR, the use of the magnetometer for a reference is not a practical solution. Magnetic referencing suffers in the OR due to external magnetic field disturbances, (e.g., during cauterization). [107]. For this reason, the IMU tracking presented here is not a suitable solution in the OR. A solution to this is to use an external tracking system for a reference in combination with the IMU. The Oculus Rift (Oculus VR, USA) uses inertial sensors in combination with an external IR tracking system that provides a reference heading. The headset is also a solution to display the stereoscopic 3D images to the surgeon, so it was chosen to replace the IMU in the previous sections. The Oculus Rift uses an infrared sensor that tracks markers on the Rift headset, providing the fixed reference point instead of the magnetometer (Figure 4.14).

There were some limitations with the hardware, in that the Oculus only works on Windows Platforms. The Rift also needs a fairly powerful computer for smooth operation. To meet the

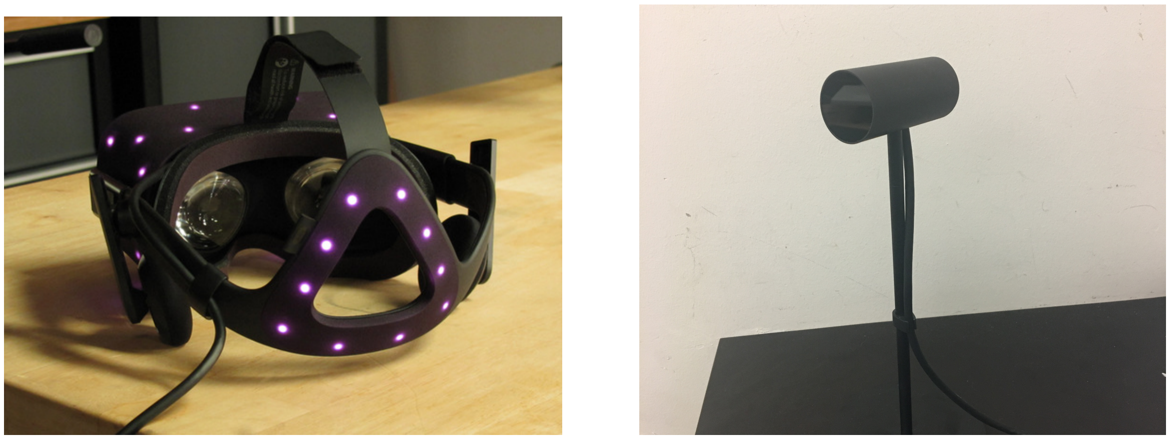

(a) Infrared LEDs on Headset (b) Infrared Camera

Figure 4.14: Oculus Rift external head tracking

requirements, the headset was connected to a Windows PC (3.4 GHz Intel i7 CPU, GTX1060 graphics card and 16 Gb RAM), and was set up as a TCP/IP server that publishes the head tracking data in quaternion form to the ROS network. On the Linux platform, these data are subscribed to, where the yaw and pitch orientation extracted, and the desired joint angles are computed before sending to the robot through ROS. Using the Rift does not alter the method of controlling the ECM. The headset provides the orientation of the head in quaternion form, and the yaw and pitch angles are extracted from this. These angles are used to compute the desired joint angles in the same manner as the IMU.

4.4 Stereo Headset

With the master console being replaced by the Leap Motion controllers, there is no longer a stereo viewing system. The initial testing in Chapter 3 was done with a direct view of the tools and objects of interest. This would not be possible in a real scenario, where the tools are inserted into the body cavity, so the endoscopic camera is needed. To replace the viewing system, a head mounted display system was developed that takes the stereo signals from the endoscope and displays them to the user in a VR headset (Oculus Rift).

One of the main benefits of the da Vinci system over traditional laparoscopic surgery is the 3D stereo vision that the surgeon is immersed in. It has shown to improve performance in various studies [9]. This works by sending a separate image to each eye, and the brain then interprets it as a 3D image. VR headsets also work in this way, where a slightly offset image is sent to each eye to give the user an immersive 3D experience. The Oculus Rift has gained popularity as a consumer grade VR headset, and has shown promising results in steresoscopic 3D medical imaging [108]. This section outlines the steps for taking the video feed from the da Vinci endoscope and displaying it into a VR headset, Oculus Rift (Oculus, USA). This is achieved with Open Graphics Library (OpenGL), and the Oculus Rift SDK.

The Oculus Rift works by rendering a different scene to separate LCD screens in front of each eye. Each eye views the scene through a custom lens which magnifies the image to provide a wide field of view. However, this lens distorts the image significantly, resulting in a pincushion distortion if the images were displayed as original (Figure 4.15a). In order to counter this distortion, post processing must be done on the rendered scenes in order for the user to view the images undistorted. This post-processing is done with an equal and opposite barrel distortion (Figure 4.15b). By pre-

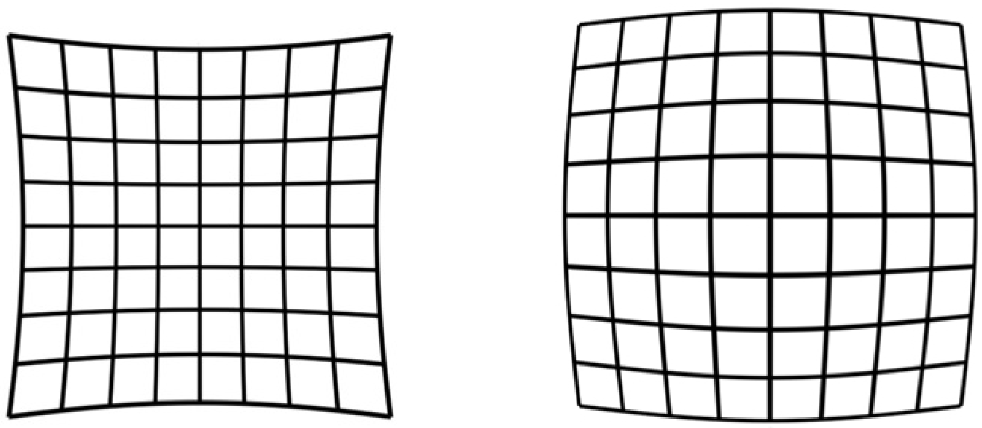

(a) Pincushion distortion (b) Barrel distortion

Figure 4.15: Distortion created by the Oculus Rift lens in (a); and the distortion pre-applied in (b)

distorting the image before displaying on the screen, the two cancel each other out resulting in an undistorted image seen by the user. In order to get the correct stereoscopic 3D effect, the horizontal position of the images is important, and varies between users depending on their inter-pupillary distance (IPD). The average IPD in humans is 65 mm [109], and the Oculus lenses can be shifted between 58 mm and 71 mm. The device uses two LCD screens placed side by side with a physical width of $w_{lcd} = 74.88$ mm [110], therefore the center of each screen is positioned w_{LCD} apart. This does not necessarily match up with the IPD of the user, so the image has to be shifted towards the center by h_m in mm:

$$h_m = \frac{w_{lcd} - w_{IPD}}{2} \tag{4.17}$$

This shifting can be done in pixels h_p:

$$h_p = W \cdot \frac{h_m}{w_{lcd}} \tag{4.18}$$

where W is the horizontal resolution of the LCD screen. Before starting the application on the headset, the user's IPD is found using the calibration provided with the Oculus software. This calibration has the user look at a cross in the middle of the screen, and adjust the lens spacing until the lines are focused. Once focused, the value of the lens spacing in mm should match the users IPD, and this value is saved and used to calculate the horizontal shift required.

In order to acquire the digital video from the left and right camera from the endoscope, a pair of USB framegrabbers, Epiphan DVI2USB (Epiphan Systems, Ottawa, Canada) were used. They are self-powered external framegrabbers that capture high resolution video at up to 60 Hz. The stereo endoscope used with the da Vinci has a resolution of 640 x 480 and a sampling rate of 30 Hz, making the Epiphan a suitable solution. The framegrabbers each capture a single image from the video feed every 30 ms, and saves the image in RGB format as an array of 8-bit integers. The processing of the image is done in OpenGl, which applies the distortion and shifting, and renders to final image to the headset.

The Oculus Rift needs a refresh rate of 90 Hz in order for the video feed to not be choppy (i.e, strobe like effect), so a new frame has to be sent at this rate. In order to achieve this, separate threads were created for the headset rendering, and frame grabber capturing. This ensures that the rendering can remain constant at 90 Hz, and not get blocked waiting for a new capture event from the Epiphan. In this case, the same frame will be rendered multiple times, but this allows for smooth video for the user.

An application was created which connects the Oculus Rift and framegrabbers, performs the framegrabber capture, and renders the image to the headset. The application starts by initializing the Oculus headset, and the framegrabbers. Once the framegrabber threads are initialized, they enter an endless loop, polling for a new capture event. When a new capture is available, the image is loaded into a frame variable that stores the data as an 8-bit RGB. The rendering thread loads the image from the frame capture thread, performs all the OpenGl functions, and renders the image to the headset. This occurs at a fixed rate, no matter if a new image has been captured or not. Figure 4.16 show a flowchart of the two threads.

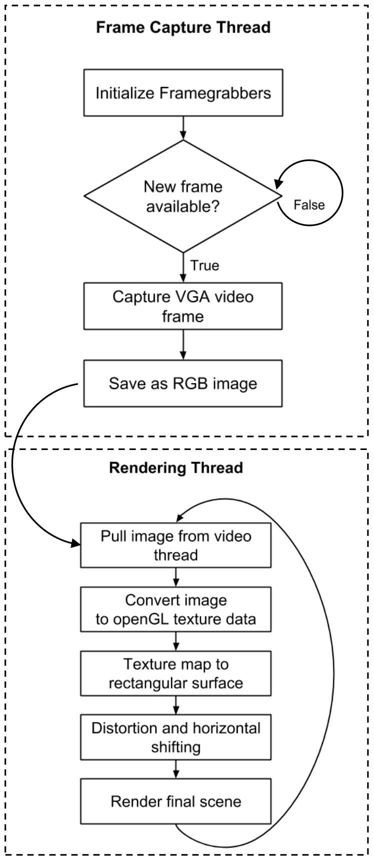


Figure 4.16: Flowchart for Oculus Rift rendering

Since the field of view (FOV) of the endoscope doesn't match up with the Oculus' FOV, the image cannot be used to fill the entire screen for the headset. This would result in a very distorted image, and cause issues with the stereoscopic view. Instead, the image must be scaled down to the percentage of this FOV to ensure that it is not distorted or cropped. The image is instead placed as a floating rectangular screen, the same distance from the eyes as in the da Vinci stereo viewer. This is to ensure the correct angle of convergence between the user's eyes and the image. According to [111], the average endoscope-to-target distance is 38 mm, and using an endoscope with lens spacing of 5.5 mm, this creates an angle of convergence with the tissue of 8.3 degrees. The average distance between the eyes is 65 mm, so to create the same angle of convergence, the image is placed a distance of 45 cm from the viewpoint in the virtual window. The images from the framegrabbers have a resolution of 640×480, so the final render target is set up to this aspect ratio, and scaled to size to render the image at the desired distance. The image is no longer able to fit the entire viewing screen, but instead the image is placed as a floating rectangle in the middle of the viewing area. Shown in Figure 4.17, the images that are rendered have a barrel distortion,

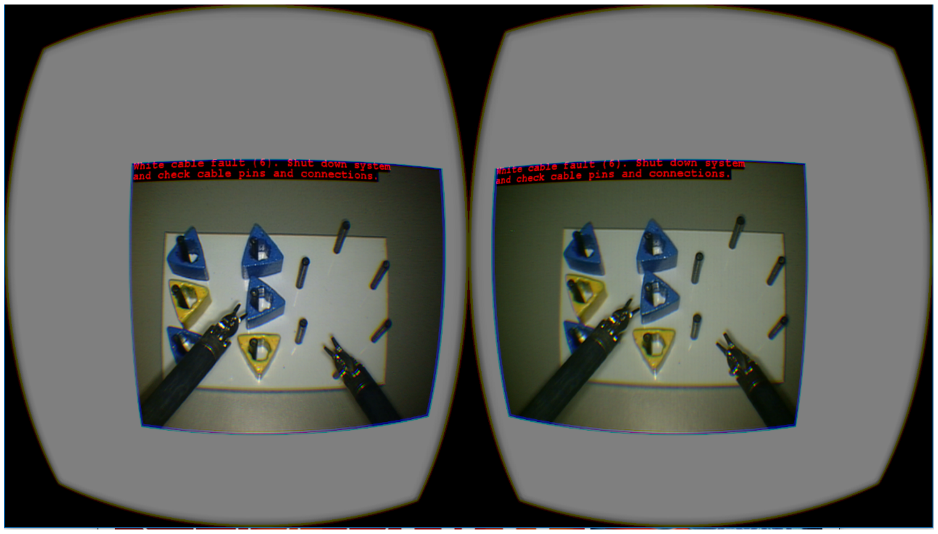

White cable fault (6). Shut down system
and check cable pins and connections.

Figure 4.17: Rendered scene to the Oculus Rift.

and are shifted into the centre of the viewing area. This loses the effect of the full immersion found in VR applications, however still provides the stereoscopic 3D view for the surgeon, in a floating screen. Inititial testing of the system gave a very similar feel to the stereo viewer in the da Vinci console. It is as if a 3D screen is floating in front of the user's eyes.

When testing this setup, it was found that viewing the scene with a plain background resulted in motion sickness. This is a commonly reported issue in VR applications, if not set up properly [112]. When one moves their head, the brain expects the view to move in the background according to the movement of the head. With just a plain background as in Figure 4.17, there is no reference point for the user and motions sickness can occur. To correct for this, a virtual room was placed behind the user using OpenGl textures. The generated walls and floor remain fixed, and as the user rotates their head, the image is updated to give a slightly different view. The floating screen remains in the middle of the screen at all times, moving around the room as the user rotates their head. Figure 4.18 shows the same floating screen, with a virtual room in the background. Having this scene in the background gives the user a fixed point as a reference, and has helped with the motion sickness issues.

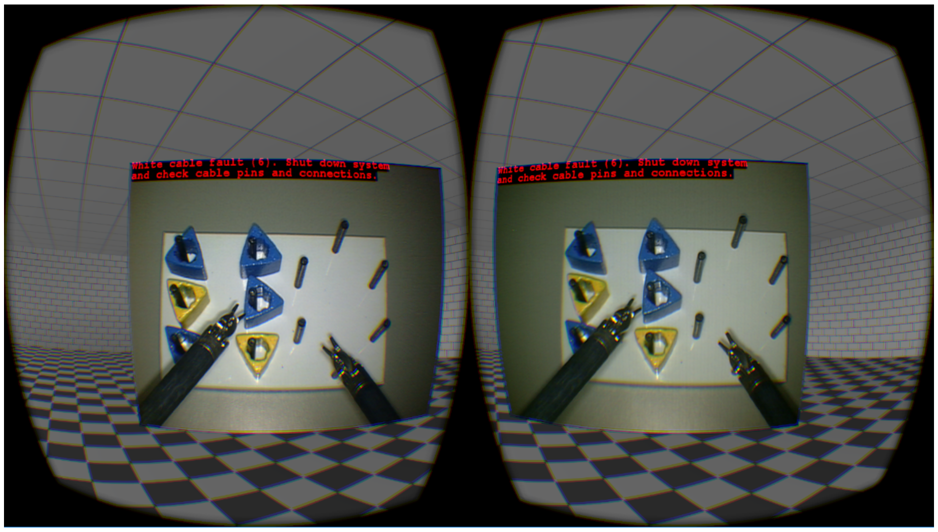

White cable fault (6). Shut down system
and check cable pins and connections.
White cable fault (6). Shut down system
and check cable pins and connections.

Figure 4.18: Rendered Scene to the Oculus Rift with virtual room

In future work, and endoscope with a wide FOV could be used that would give the user a more immersive experience in the VR headset, allowing the image to fill the entire viewing area.

4.4.1 Audio

The last function of the VR headset is for providing the user with auditory feedback. The headset contains built in headphones that are able to provide the user with audio in the VR world. When wearing the headset, the surgeon can no longer see their hands, not knowing where they are relative to the leap sensors. To address this, a virtual boundary is placed as defined by the workspace in Chapter 3. The boundary is defined as 0–250 mm in the x direction, -100–100 mm in the y

direction, and 100–400 mm in the z direction above the controller. Auditory feedback is used to alert the user when their hands are too close to the boundary. The position of the wrist of the hands are tracked with the Leap Motion controller and compared with the boundary set up in the x, y, and z planes. As the hand approaches one of these boundaries, an audible alert tells the user which hand, and what boundary they are approaching. This allows the user to clutch and move their hand back into the centre of the workspace. This communication between the two systems is also done through the ROS TCP/IP server, the same way the Oculus broadcasts its tracking information, but in the opposite direction. Each hand in compared to the position of the boundary, and if within a certain threshold, a message indicating which boundary the hand is approaching is broadcast on ROS. The windows client then processes these data and plays the corresponding audio file to alert the user.

With the headset connected the the Windows PC, the third Leap Motion controller is connected to this PC as well, and the hand data sent to the Linux PC over the same TCP /IP connection. The system diagram for all the hardware can be seen in Figure 4.19. The main Leap Motion sensor is connected to the Linux PC, and tracks both hands. Then the other two sensors, Leap2 and Leap3 are connected to the Virtual machine, and Windows PC respectively, tracking only one hand each. There is bi-directional communication between the two PCs, for sending the head tracking and Leap data to the Linux PC, and the workspace boundary check data from the Linux PC to the Windows PC. The Linux PC runs the main node which communicates with the dVRK controllers for positioning the joints of the manipulators.

4.5 Conclusions

A stereo headset was used to provide the user with an immersive stereo view, without the need for the large stereo viewer in the da Vinci. This headset is also used to provide a novel method of control of the endoscopic camera, possibly providing the user with a more intuitive method of controlling the camera during surgery. Two methods of control are proposed, one where the user is able to look around at all times, and the other where a pedal is used for control. It is still to be determined which method will be more intuitive for the user, and future studies will be conducted.

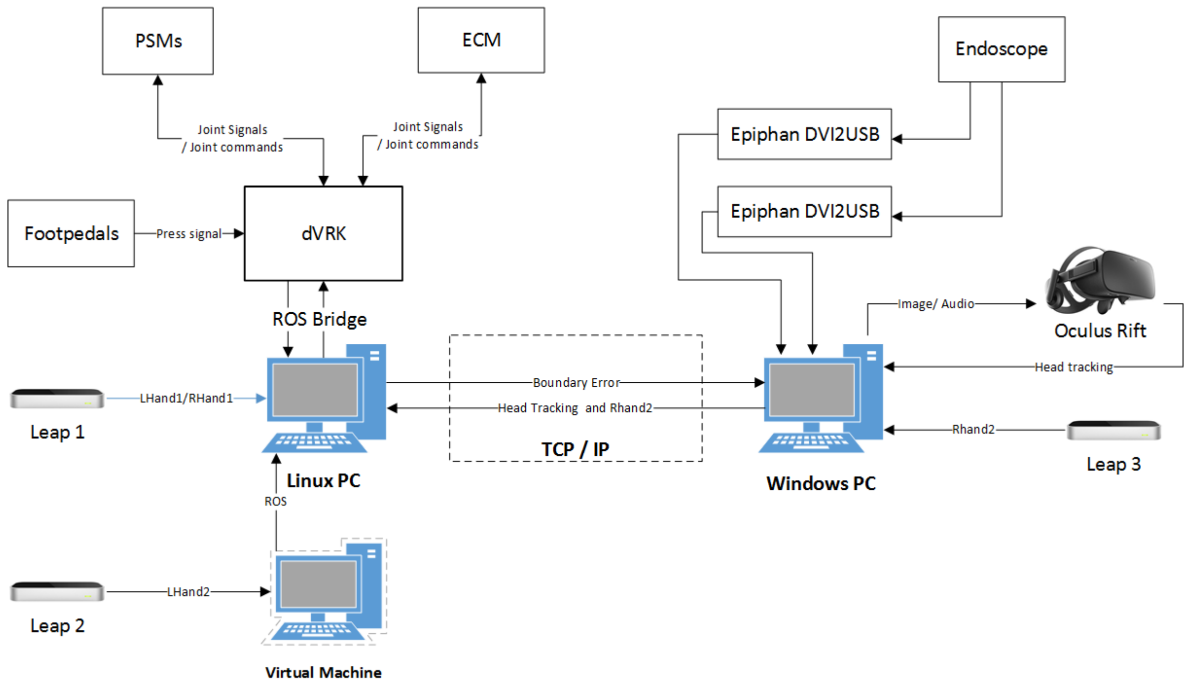


Figure 4.19: Full hardware setup and information flow

Chapter 5

Validation

This chapter outlines the testing and validation done on the system in Chapter 3 and 4, although the camera was held still during the tasks. A separate study will be conducted for the validity of the endoscope control. A pilot study was done to assess the hand tracking system in its preliminary stage. This study is to assess the performance of the system, using the original da Vinci master manipulators as the control. The original system will be mentioned as 'classic' throughout this study. This study is also used to iron out any issues with the methods and set up before beginning a larger study with more participants.

5.1 Methods

For this pilot study, a total of 4 participants completed a surgical training task: peg transfer. 3 subjects were considered novices, and 1 considered an expert. The criteria for novice was a user with no medical background, e.g., engineers, and an expert is a surgeon with a certain amount of experience in robotic surgery. It must be noted that all subjects had some prior experience using the classical da Vinci system. The peg transfer task is part of the surgical training curriculum, outlined in the Fundamentals of Laparoscopic Surgery (FLS). This task was chosen as it is a fairly simple task suitable for novice users, however it involves a certain range of motion not seen in a the pick and place task in Chapter 3. The task requires a fair amount of dexterity, and involves using both hands, passing objects between them. It was chosen as the task to assess the system since the main issues are with occlusion, and this task can assess the performance when the hand is rotated through the range of motion necessary for the task. The subjects were given instructions on how to complete the task, as well as time to practice on both systems (5 minutes on each). Each subject was asked to complete the tasks three times on both the classic da Vinci, and the Leap system. The order of the systems were randomized. The expert did not complete the trial on the classic system due to time constraints.

Task: Peg Transfer

The peg transfer task, seen in Figure 5.1 involves picking up objects with both hands, and tests dexterity and hand eye coordination, as well as depth perception. The task begins with the

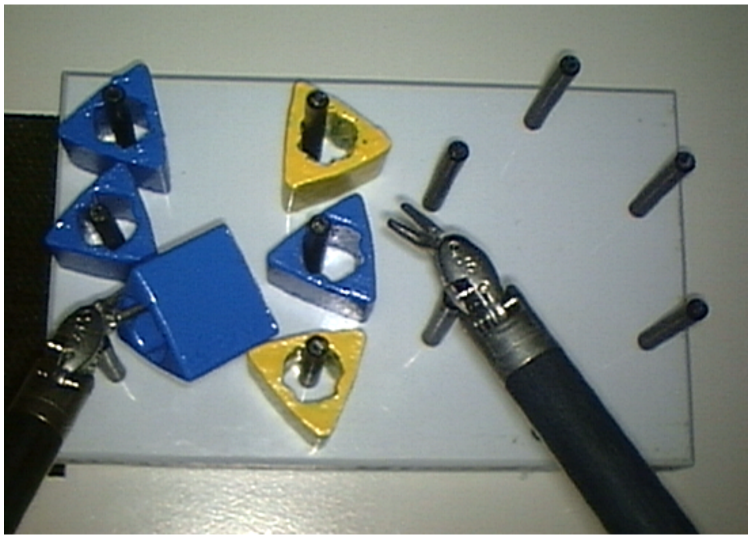


Figure 5.1: Peg transfer task

6 triangle ring objects aligned on the same side of the board as the subject's non-dominant hand. The subject then has to pick up the objects with their non-dominant hand, and transfer the peg mid-air to the dominant hand before placing it on the pegs on the opposite side. Once all six objects have been placed, the process is reversed, transferring all six objects back to the original starting position. The number of drops was recorded, and the time to complete the task, as well as the number of times the clutch pedal was used to reposition the hands. Video of the endoscope feed was recorded, as well as video of the user's hands during the task. Drops of the objects within

the field of view do not affect the time score, but drops outside do. If this occurs, the subject was told to continue with the next object, and leave the dropped one, and a penalty of 10 seconds is added. The subjects were also asked to respond to a set of statements, ranking their opinion on a 5 point Likert scale ranging from strongly disagree to strongly agree. The statements were: "the system was easy to use","the system was comfortable to use", "the control of the instruments was intuitive", "I felt that the instruments matched my hand movements accurately", and "I would be able to use this system for an extended period of time". The last statement was to assess whether the user was getting fatigued using the system. Finally, they were also asked to give any additional feedback or input they had on the set up.

5.2 Results

The results of the trials are presented below, including the completion time, the number of drops, and the amount of clutching.

5.2.1 Completion Time

Task completion time results are presented in Figure 5.2 and in Table 5.1. The time to complete the task was lower for the classic system in all cases. The variation in time was fairly small with the classic system, indicating that the users completed the task in relatively the same manner in all three trials. The variation in time for all subjects was much larger for the Leap system, indicating that there was much more variability in the performance.

Table 5.1: Task completion time in seconds

	Novice1	**Novice2**	**Novice3**	**Expert**	**Mean**
Mean Time (Leap)	184 ± 52.4	253.3 ± 48.5	181.3 ± 35.8	121.3 ± 42.7	185 ± 44.9
Mean Time (Classic)	111.0 ± 13.1	168.7 ± 10.1	103.7 ± 7.8		127.7 ± 10.3

5.2.2 Drops

The number of drops recorded during the tasks are presented in Figure 5.3. A drop can be considered an error, and an additional indicator of the performance or precision of the user. The

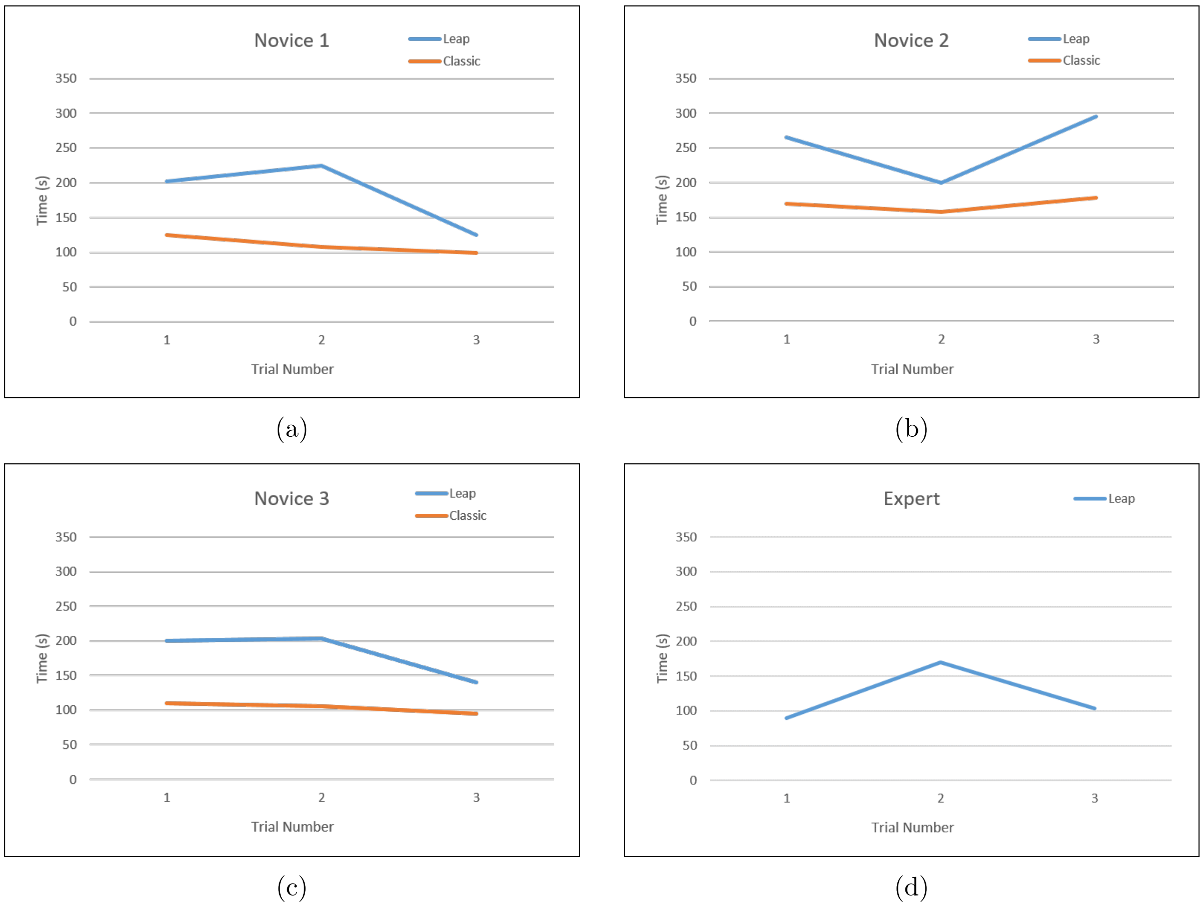
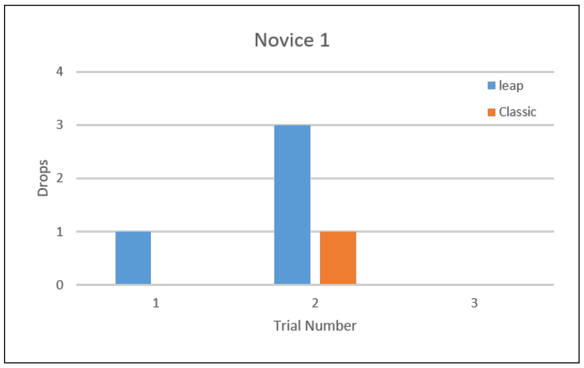


Figure 5.2: Task completion time of 4 subjects comparing both systems

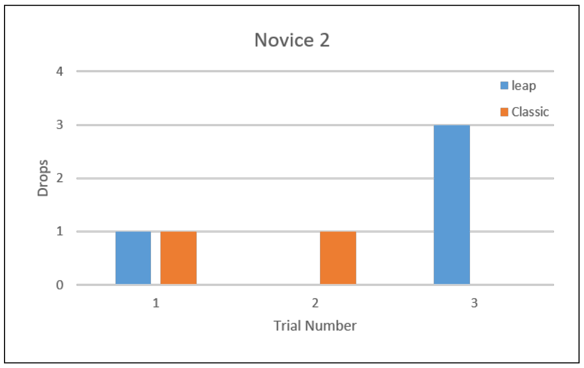
task completion time correlated fairly well with the number of drops that occurred. This makes sense, as it takes additional time for the user to pick the object up again. When analysing the video recordings of the tools and the user's hands, the most common place that the object was dropped was when the subject was passing the object between the tools. The tool would sometimes open when not expecting it, while the user's hands were still attempting to grasp the object.

5.2.3 Clutching

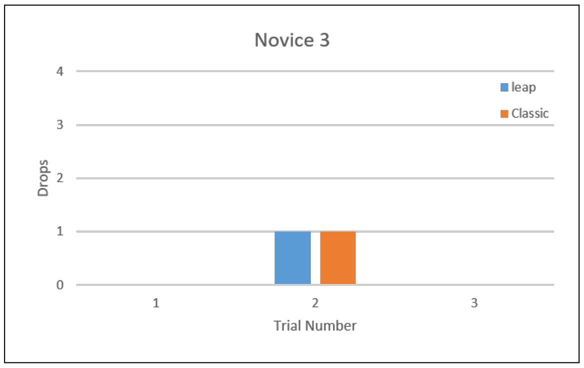
The amount of time the clutch pedal was used during the task was recorded, and results are shown in Table 5.2, with the mean values in Figure 5.4. Ideally, the clutch pedal should not be pressed at all, as the workspace range is large enough to reach all of the targets.



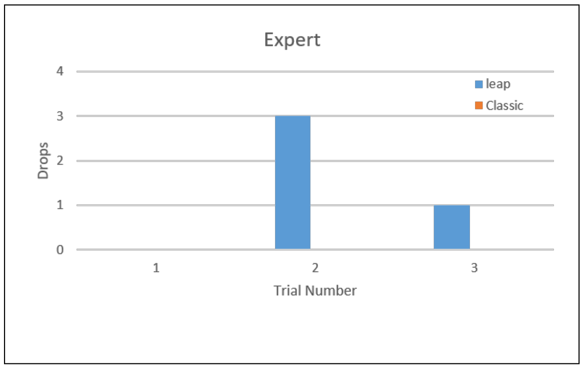
(a)



(b)



(c)



(d)

Figure 5.3: Number of drops comparing both systems

Table 5.2: Number of clutches during peg transfer task

	Novice1			**Novice2**			**Novice3**			**Expert**		
Trial Number	1	2	3	1	2	3	1	2	3	1	2	3
Leap	3	6	1	3	1	4	4	4	2	6	9	7
Classic	1	0	0	0	0	1	2	0	1			

5.2.4 Questionnaire

The results of the questionnaire are presented in Figure 5.5. The rankings are as follows: 1: strongly disagree, 2: disagree, 3: neutral, 4: agree, and 5: strongly agree. All subjects ranked the classic system higher in all categories. The Leap system was ranked the highest in comfort, and ability to use for an extended period of time. The categories that were the worst are "did not restrict hand movements" and "followed hand movements accurately". The users ranked the two

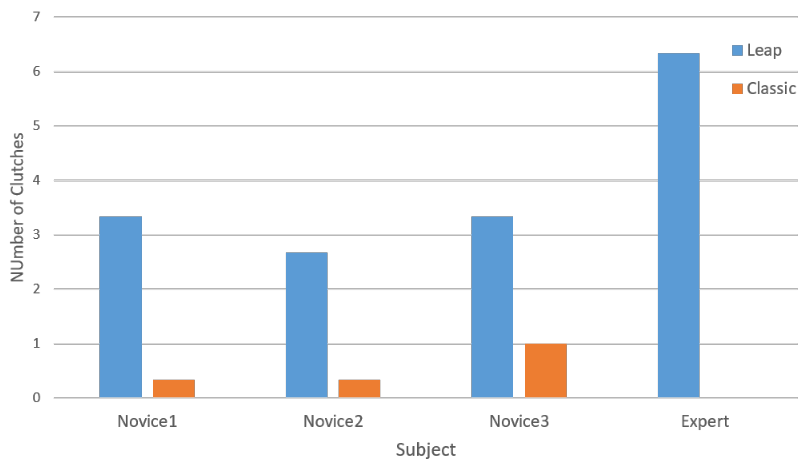
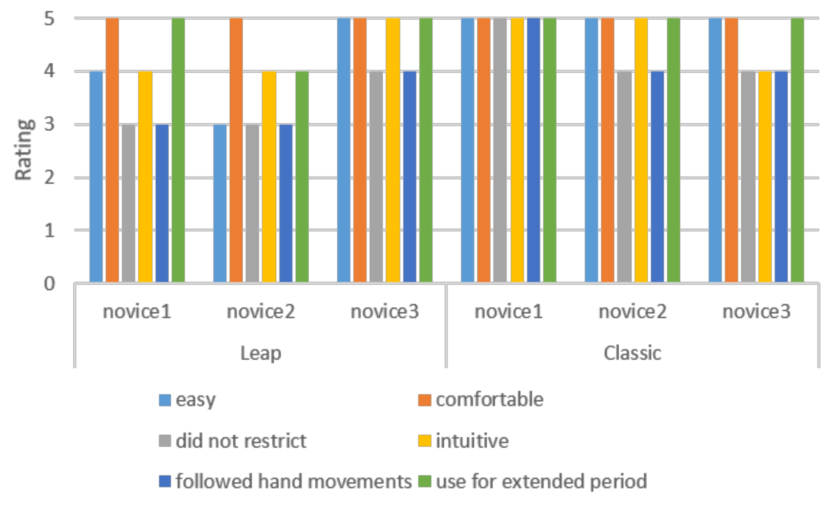


Figure 5.4: Mean clutching amount: leap vs classic system

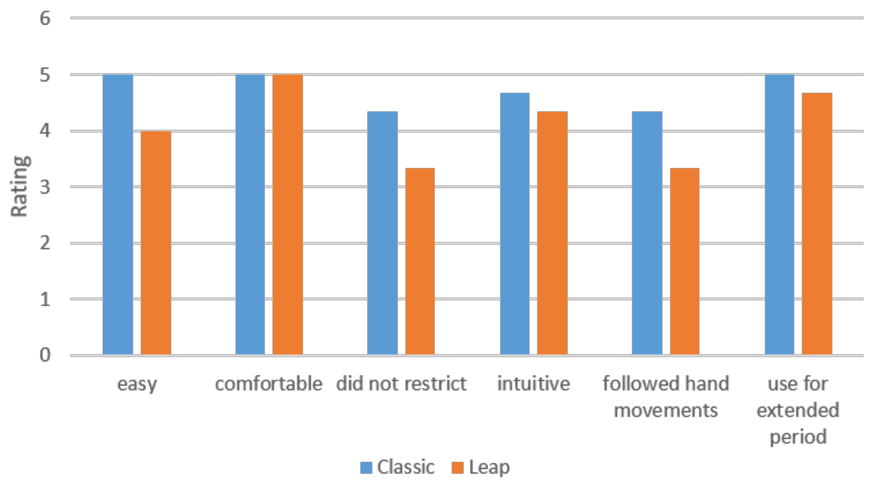
systems fairly close in ease of use, comfort, intuitiveness, and ability to use for an extended period. Additional feedback was given from two novice subjects: Novice 1 commented: "A longer learning period is required. I think the performance can be significantly increased with more practice on the leap system", and "not used to grasping things without feeling anything" Novice 3 commented: "I felt that I was getting the hang of it by the third attempt, it was difficult to keep track of where my hands are".

5.3 Discussion

With the small sample size it is difficult to come to any conclusions right now, however, the trials did show where improvements in the set up can be made. The system was functional, and all users were able to complete the task without dropping the objects outside the field of view. The mean completion time for the Leap was 185 seconds versus the classic da Vinci time of 127 seconds. However, the range of times for the Leap system was much larger for each user, where the average range was 45 seconds, versus 10 seconds on the classic. This was due to the fact that some trials went very smoothly, while others did not, due to issues with the set up. In several trials, the user was able to get through the entire trial very quickly with no errors. In other trials, drops of the object occurred due to unwanted opening of the grasper. This caused them to need to pick the



(a) Individual Scores



(b) Mean Scores

Figure 5.5: Questionnaire Scores. (a) individual rankings; (b) mean scores

object back up, and potentially cause a loss of concentration. In the trials that were completed with no drops or clutching, the performance was fairly comparable to the classic system.

The video of the user's hands during the trials showed that the drops were occurring in the same point for all users. This was when the hands were very close to one another. What was not addressed in the original design, was the aliasing that happens from the opposite hand. When the left hand is directly behind the right from the line of sight of the right sensor, the fingers are not tracked with the same accuracy as described in Chapter 3. Even with the hand in a suitable orientation for tracking, there were issues with the grip control, presumably due to being close to the other hand.

The amount of clutches with the Leap system was much higher than with the classic system. The reason for this was due to the subjects reaching the edge of the workspace before reaching the object they are attempting to pick up. In certain trials however, the user was able to complete the task without clutching at all, indicating that the workspace was large enough to complete the task. The issue in this case was the starting position for the hands. For this task, the starting position of the tools was consistent, with the surgical tools positioned in the center of the task board. The MTMs of the da Vinci start in the centre of their workspace, allowing the required range of motion for the task. On the other hand, the Leap system has users place their hands above the workspace, and then the tools match to their hand orientation. If the hands are placed in the center of the workspace for each hand, the task can be completed without clutching. If the hands are positioned off-center to begin, it makes it not possible to reach the edges of the peg board, resulting in the need to clutch. Therefore, the comparison between the two systems is not completely fair, as the starting position predicts how often the user will need to clutch again. This is a more advanced skill which takes practice to master.

With the classic da Vinci the user's hands are also mechanically contained in the workspace, and cannot physically move past it. In this system, an auditory alert lets you know when your hands are too close or too far from the sensor, but can still physically move past, and some users did not respond immediately to the warning, causing a degradation in tracking as they continued outside the boundary.

This task involves picking and placing objects at different distances from the camera, and tests the user's depth perception. None of the users reported any loss of depth perception using the head mounted display compared to the da Vinci stereo viewer. Additional trials can be done using the same master interface, comparing the headset with the stereo viewer in the surgeon console, to test for any variability in depth perception between the two.

5.3.1 Questionnaire and Additional Comments

The consensus from the three novice users, was that the control was intuitive, and comfortable, but the tools did not match the movements of the hand perfectly. Every subject had issues with the system at one or more points, where the tools did not match the movement of the hand well. This can be explained by the loss of accurate tracking when the hands are too close together. The purpose of the questionnaire was to assess the subjects' opinion on the system, and due the issues at the point of passing the object, this may negatively influence the score for all categories. The scores may not accurately reflect the performance of the system when used in the correct manner (e.g., having hands centred to begin).

The comments the users had on the system indicated that due to the nature of the system being new (grasping without the sense of touch, and not being able to see the hands), that it might take more time to become accustomed to this method of controlling the tools. All subjects had prior experience using the classical da Vinci, and is much more 'normal', as in it is close to other interfaces such as a joystick. Since this is a novel system, the sensation is completely foreign, and may take more time to get used to it. The common occurring theme was that the control felt intuitive, but not knowing where their hands were when wearing the headset proved to be difficult in keeping their hands in the workspace.

5.3.2 Future Trials

Before additional trials are completed, addressing these issues is important, in order to ensure that the users are using the system to the maximum potential. One possible solution for the interference of the opposite hand would be to place a barrier between the hands in the middle of the workspace. Each hand would have a separate tracking space, independent of the other. A barrier would have the same problems with aliasing however, so it would need to be made of a material that absorbs infrared light, to not interfere with the tracking of the hand. There has been some success with this using a material called Duvetyne, used in the film industry for making dark backdrops [113]. This would ensure that the hand is tracked well with no external objects behind causing tracking issues.

Another possible solution is to have a calibration routine at the beginning before taking control of the tools. The user would have to place their hands such that they are hitting a target on the virtual screen in front of them, once the target is hit, control will begin. This would ensure that the hands are centred in the workspace, so the optimal starting position is found, maximizing the area to each side. Allowing the surgeon to see where in the workspace their hands are during the task is also a possible addition, so it is not a surprise when their hands reach the edge of the workspace. This can be done with an overlay on the side of the surgeon's screen, letting the surgeon know where their hands are relative to the boundary. Before additional trials, these issues will be addressed, so the comparison between the two systems is more fair. Additional subjects who have not used either system will also be recruited to ensure the experience level is the same, to get a better understanding of the learning curve for both systems.

5.4 Additional Complex Trial

Due to the performance of the system being sensitive to small details such as starting position, a single subject was given additional time to train with the system. After practising for a total of 1 hour on both systems, and becoming more familiar with the systems, the user completed an additional more complex task. The user had an understanding of the issues with hands being close together, and was instructed to begin the task with the hands set farther apart so the hands never interfered with one another.

After practising with the system, the additional task tested was suturing and knot tying. This task involves placing a suture through two marks in a penrose drain (Figure 5.6), and then tying three throws of a knot, closing the slit in the penrose drain. The needle has to be switched between the hands between each knot, to ensure that it is being tied with the opposite hand each time. This task is the most complex of the FLS tasks, and requires a great deal of dexterity with the tools, requiring repositioning and regripping of the small needle many times. It was chosen to assess the performance of the system when more dexterity is needed. In order to tie the knot, the user has to wrap the suture around one of the tools before pulling the other end of the suture through that created loop. This involves a great amount of precision and takes practice to master

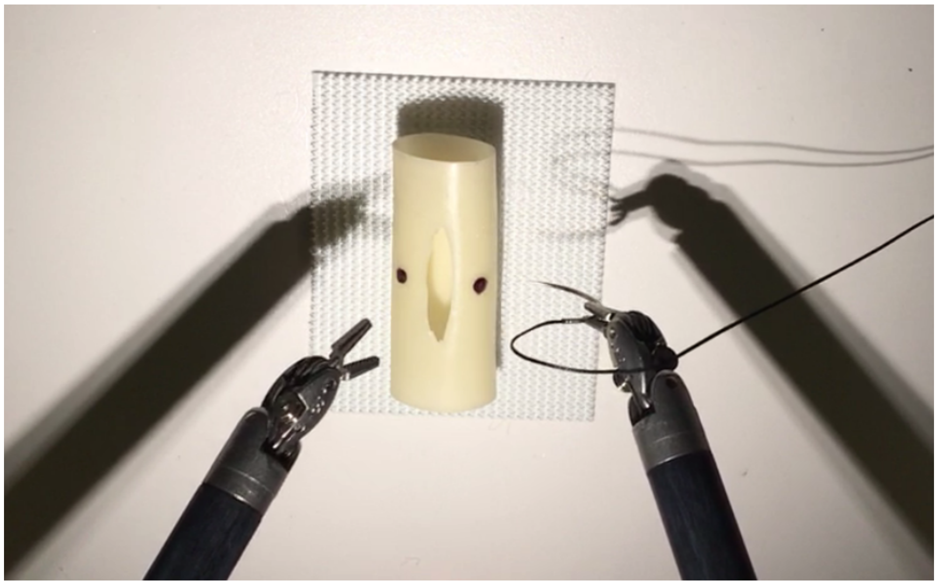


Figure 5.6: Suturing Task

the skill. The time to compete the task is measured, and penalties are assessed from deviations of the finished suture from the marks in the penrose drain, or if the knot slips or comes apart under tension. The amount of times the needle was dropped was also assessed.

5.4.1 Results

The subject completed the task 3 times on both systems. The time to complete the task is shown in Table 5.3.

Table 5.3: Suturing Trial

	Classic			**Leap**			**Mean**	
Trial Number	1	2	3	1	2	3	Classic	Leap
Completion Time (s)	165	128	135	151	160	144	**142.7**	**151.7**
Missed Targets	0	1	0	0	1	0	0.33	0.33
Needle Drops	0	0	0	0	1	0	0	0.33

The time to complete the task on both systems was comparable. With the average time on the classic system of 143 seconds, and the leap system of 152 seconds. The accuracy of hitting the marks was also the same, with one miss on the exiting mark on both systems in one of the three trials. The leap system had one drop of the needle and the classic system did not. The knots were tied correctly in all 6 cases.

5.4.2 Conclusions

The pilot study of the system in the initial stages has shown that the users are able to successfully complete a surgical task requiring a large range of motion of the hand. It has also shown areas that need to be improved. Given time to practice with the system, it has been shown that the user was able to control the tools with better performance. The subject was able to successfully complete the task in all three cases, with times comparable to the classic system. This has shown that it is possible to achieve the dexterity needed for a complex task such as knot tying with the system created. The main reason for the improvement is the understanding of the limitations of the system, and ensuring that the hands start in the correct position. Knowing that the hands needed to be farther apart, the subject was able to complete the task all three times with no drops, in a similar time as the classic system.

This has shown promising potential in the system, and also has indicated areas that need improvement. The main issues being with the setup of the user during the task. Providing the user with a way to ensure their hands are optimally placed is needed. The hypothesis is that ensuring that the user starts in the correct position will improve the task performance.

Chapter 6

Conclusion and Recommendations

This chapter concludes the research presented in this thesis on the topic of novel methods of interaction with the da Vinci surgical robot. A brief summary of the work done, and possible extensions on this research for future work are presented.

The work presented in this thesis was aimed at addressing some of the current limitations in RAMIS, specifically dealing with the human machine interface for the da Vinci robot. A literature review was performed to show prior attempts at solutions and provide a background and motivation for this research. The da Vinci surgeon console is very large and expensive, and disassociates the surgeon from the patient. The master manipulator is a mechanical device that the surgeon must contact to control the surgical tools, preventing the smooth transition between traditional and robotic surgery. There has been extensive research into different methods of control for robotic surgery, and the aim of this research was to develop and assess a novel system for interaction in a non-contact manner. Highlighted was the method for controlling the robot using infrared sensors that track the position of the hands in 3D space. The natural motions of the hand were then translated to the tools of the da Vinci. Preliminary testing showed the capability of the system, however, occlusion issues were shown when rotating the hand in motions needed for more complex tasks. The solution using multiple sensors from multiple view points was shown, and the tracking performance has improved greatly. The method of replacing the stereoscopic viewer in the surgeon console is shown, using a consumer VR headset to display the surgical site to the surgeon. A novel method of controlling the camera was proposed, using the motion of the head. Preliminary testing showed positioning of the head with acceptable precision. A pilot study for the hand tracking and VR headset setup was run, indicating the performance of the system can be comparable to the da Vinci, however, current issues will need to be addressed to maximize the potential of the system.

6.1 Contributions

This work shows the development of novel methods of interaction with surgical robotic systems. The specific contributions of this work are as follows:

- A system for teleoperation of the da Vinci surgical robot using natural hand and head motion was developed. The system used consumer infrared sensors to track the hands in 3D space, and inertial sensors to track the head motion. A full teleoperation platform was developed using ROS, allowing the user to control the surgical tools and camera system. The system's surgeon console has been replaced with much less expensive components, and allows the surgeon a more natural interface.

- A solution to occlusion issues with vision based tracking has been shown. Using multiple sensors at different viewpoints, the tracking accuracy of the Leap motion controller has been improved greatly for a larger range of motion. The hand can successfully be tracked with an accuracy of less than 5 mm for the entire range of motion of the hand during complex surgical procedures.

- A novel method of controlling the camera arm was proposed. The motion of the head was used to move the camera in a natural and intuitive fashion using inertial sensors. Preliminary testing has shown that this method of control can be done with fairly high precision, and further testing is ongoing.

- A consumer VR headset was used to display the live video feed from the da Vinci endoscope to the surgeon in a stereoscopic 3D viewing screen. This is the first time a VR headset has been used in this fashion for displaying live video with the da Vinci Surgical Robot.

- A pilot study has been performed, showing that the system is capable of completing complex surgical tasks. The system was developed using commercially available components, and highlights the performance of the device compared to the da Vinci console. This research shows the potential of vision based tracking for surgical procedures.

6.2 Future Work

Testing of the system in Chapter 5 has highlighted the immediate future work that needs to be addressed.

1. Through the trials, it was found that a better method of initializing the tools when the user places their hands in the workspace is needed. In order to optimize the trackable area, the surgeon's hands should begin in the center of the workspace. In order to achieve this, a calibration routine at the start of task can be done. By ensuring the hands are centred at the beginning, it will allow the surgeon the maximum usable range on either side of the hands.

2. A solution for self occlusion of the hand has been presented using multiple controllers. However, what was not addressed was the interference in the tracking of one hand with the opposite hand. When the second hand is directly in line with the fingers of the hand of interest, the tracking performance is much lower. To address this issue, keeping the hands separated further to begin may help, but another solution is also presented here. Placing a barrier between the hands will create a separate workspace for each hand, free of occlusions from the other. In order for this to be effective, the divider must be made of a material that absorbs infrared light, to not interfere with the tracking.

3. Users found that it was difficult to keep track of where their hands are in the workspace, and get surprised when reaching the edge. Additional information can be overlaid onto the head mounted display informing the user of the position of their hands in the workspace. This will allow the user to know where there hands are at all times, so they can preemptively plan when to clutch and move their hands back to the center.

The main focus of this work was on the development of the system, and only a preliminary trial was done. More extensive testing will be needed after the issues outlined previously are addressed in order for the system to reach its full potential. Other work to be done that may improve the system is outlined here:

6.2.1 Hand Tracking and Haptics

Using vision based tracking results in the control of the tools with no contact for the user. This is a new sensation for users, and may take additional time to get used to. It may be difficult to keep the fingers pinched at the desired distance, having to strain the muscles in the hand. Further testing will be necessary to see if this will be an issue, but one proposed solution would be to include a small spring strip along the inside of a surgical glove. This will give the user some sensation of grasping something, providing resistance to pinching. This will not be the true force they are applying to the tissue, but would be similar to the sensation found in the current da Vinci system. Other things to look into include using a mechanical device that would provide some resistance to moving, allowing for haptic feedback.

6.2.2 Headset

Using the Oculus Rift as the stereoscopic headset allows the easy implementation of additional information onto the screen. The Oculus Rift can be rendered to using various programs such as Unity Game engine or OpenGl and is open to developers to implement their own ideas. This would be useful in adding additional information for the surgeon such as force information of the tools. Since there is no haptic feedback for the user, using sensory substitution in combination with a sensorized tool may be useful in certain procedures that are senstive to the amount of force being applied.

Due to the Oculus Rift being a VR headset, where the user is placed into a virtual world, the surgeon no longer has a view of the operating room. Another possibility is to replace the VR technology with augmented reality (AR), allowing the surgeon to view the operating room while still having the same stereoscopic view of the surgical site. With AR technology, images are superimposed on the user's view of the real world. The same overlays can be placed in an AR system as in the VR headset, making this a suitable solution.

The Oculus Rift allows for a wide field of view (FOV) for immersion in the VR application.

However, in order to maintain the correct FOV in the headset, the usable screen size was limited due to the narrow FOV of the endoscopic camera. Using an endoscope with a wide FOV would enable the whole screen of the Oculus to be used, creating a much more immersive effect for the surgeon.

6.2.3 Testing and Validation

When the issues are addressed to ensure the maximum potential of the system, further trials will be run with a much larger sample size. Novice users who have not used either system will be recruited to ensure the same amount of experience on both systems. This will allow the assessment of the learning curve for the device. Expert surgeons will also be recruited to receive the feedback from surgeons with experience on the system. The trials will include more tasks, including beginner, intermediate and advanced tasks that will asses the performance of the system at different levels of difficulty.

Additional trials are to be run for testing the performance of the endoscopic camera. The proposed trial would have subjects move the camera to center on targets and hold before moving to the next target. The time to complete the task and the accuracy would be measured, as well as feedback for how comfortable and intuitive the system is.

References

[1] D. F. Del Rizzo, W. D. Boyd, R. J. Novick, F. N. McKenzie, N. D. Desai, and A. H. Menkis, “Safety and cost-effectiveness of MIDCABG in high-risk CABG patients,” in *Annals of Thoracic Surgery*, vol. 66, no. 3, 1998, pp. 1002–1007.

[2] C. Preusche, T. Ortmaier, and G. Hirzinger, “Teleoperation concepts in minimal invasive surgery,” *Control Engineering Practice*, vol. 10, no. 11, pp. 1245–1250, 2002.

[3] K. H. Fuchs, “Minimally invasive surgery,” *Endoscopy*, vol. 34, no. 02, pp. 154–159, 2002.

[4] A. G. Gallagher, N. McClure, J. McGuigan, K. Ritchie, and N. P. Sheehy, “An ergonomic analysis of the fulcrum effect in the acquisition of endoscopic skills,” *Endoscopy*, vol. 30, no. 07, pp. 617–620, 1998.

[5] A. G. Gallagher and R. M. Satava, “Virtual reality as a metric for the assessment of laparoscopic psychomotor skills: Learning curves and reliability measures,” *Surgical Endoscopy and Other Interventional Techniques*, vol. 16, no. 12, pp. 1746–1752, 2002.

[6] C. D. Smith, T. M. Farrell, S. S. McNatt, and R. E. Metreveli, “Assessing laparoscopic manipulative skills,” *American Journal of Surgery*, vol. 181, no. 6, pp. 547–550, 2001.

[7] A. G. Gallagher, R. Karen, M. Neil, and J. McGuigan, “Objective psychomotor skills assessment of experienced, junior,and novice laparoscopists with virtual reality,” *World Journal of Surgery*, vol. 25, no. 11, pp. 1478–1483, 2001.

[8] A. T. L. Ng and P. C. Tam, “Current status of robot-assisted surgery,” *Hong Kong Medical Journal*, vol. 20, no. 3, pp. 241–250, 2014.

[9] K. Moorthy, Y. Munz, a. Dosis, J. Hernandez, S. Martin, F. Bello, T. Rockall, and a. Darzi, “Dexterity enhancement with robotic surgery.” *Surgical endoscopy*, vol. 18, no. 5, pp. 790–795, 2004.

[10] F. Corcione, C. Esposito, D. Cuccurullo, A. Settembre, N. Miranda, F. Amato, F. Pirozzi, and P. Caiazzo, “Advantages and limits of robot-assisted laparoscopic surgery: Preliminary experience,” *Surgical Endoscopy and Other Interventional Techniques*, vol. 19, no. 1, pp. 117–119, 2005.

[11] G. Turchetti, I. Palla, F. Pierotti, and A. Cuschieri, “Economic evaluation of da Vinci-assisted robotic surgery: A systematic review,” *Surgical Endoscopy and Other Interventional Techniques*, vol. 26, no. 3, pp. 598–606, 2012.

[12] Y. J. Woo and E. A. Nacke, "Robotic minimally invasive mitral valve reconstruction yields less blood product transfusion and shorter length of stay." *Surgery*, vol. 140, no. 2, pp. 263–7, aug 2006.

[13] T. N. Payne and F. R. Dauterive, "A Comparison of total laparoscopic hysterectomy to robotically assisted hysterectomy: surgical outcomes in a community practice," *Journal of Minimally Invasive Gynecology*, vol. 15, no. 3, pp. 286–291, 2008.

[14] M. Salman, T. Bell, J. Martin, K. Bhuva, R. Grim, and V. Ahuja, "Use, cost, complications, and mortality of robotic versus nonrobotic general surgery procedures based on a nationwide database," *American Surgeon*, vol. 79, no. 6, pp. 553–560, 2013.

[15] "da Vinci Surgical System 2013." [Online]. Available: http://www.intuitivesurgical.com/products/davinci{_}surgical{_}system

[16] V. Falk, D. Mintz, J. Grunenfelder, J. I. Fann, and T. A. Burdon, "Influence of three-dimensional vision on surgical telemanipulator performance," *Surgical Endoscopy*, vol. 15, no. 11, pp. 1282–1288, 2001.

[17] J. C. Byrn, S. Schluender, C. M. Divino, J. Conrad, B. Gurland, E. Shlasko, and A. Szold, "Three-dimensional imaging improves surgical performance for both novice and experienced operators using the da Vinci Robot System," *American Journal of Surgery*, vol. 193, no. 4, pp. 519–522, 2007.

[18] Y. Munz, K. Moorthy, A. Dosis, J. D. Hernandez, S. Bann, F. Bello, S. Martin, A. Darzi, and T. Rockall, "The benefits of stereoscopic vision in robotic-assisted performance on bench models," *Surgical Endoscopy and Other Interventional Techniques*, vol. 18, no. 4, pp. 611–616, 2004.

[19] A. D. Greer, P. M. Newhook, and G. R. Sutherland, "Human-machine interface for robotic surgery and stereotaxy," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 3, pp. 355–361, 2008.

[20] G. R. Sutherland, P. B. McBeth, and D. F. Louw, "NeuroArm: an MR compatible robot for microsurgery," *International Congress Series*, vol. 1256, pp. 504–508, 2003.

[21] F. Despinoy, S. Alonso, N. Zemiti, P. Jannin, and P. Poignet, "Comparative assessment of a novel optical human-machine interface for laparoscopic telesurgery," *Information Processing in Computer-Assisted Interventions, Lecture Notes in Computer Science*, vol. 8498, pp. 21–30, 2014.

[22] L. Santos-Carreras, M. Hagen, R. Gassert, and H. Bleuler, "Survey on surgical instrument handle design: ergonomics and acceptance," *Surgical Innovation*, vol. 19, no. 1, pp. 50–59, 2012.

[23] R. Konietschke, T. Ortmaier, H. Weiss, G. Hirzinger, and R. Engelke, "Manipulability and accuracy measures for a medical robot in minimally invasive surgery," in *On Advances in Robot Kinematics*. Springer, 2004, pp. 191–198.

[24] T. Zhou, M. E. Cabrera, S. Member, and J. P. Wachs, "Touchless telerobotic surgery A comparative study," *IEEE EEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

[25] "SPORT Surgical System." [Online]. Available: http://www.titanmedicalinc.com/technology/

[26] U. Hagn, R. Konietschke, A. Tobergte, M. Nickl, S. Jörg, B. Kübler, G. Passig, M. Gröger, F. Fröhlich, U. Seibold, L. Le-Tien, A. Albu-Schäffer, A. Nothhelfer, F. Hacker, M. Grebenstein, and G. Hirzinger, "DLR MiroSurge: A versatile system for research in endoscopic telesurgery," *International Journal of Computer Assisted Radiology and Surgery*, vol. 5, no. 2, pp. 183–193, 2010.

[27] A. De Donno, L. Zorn, P. Zanne, F. Nageotte, and M. De Mathelin, "Introducing STRAS: A new flexible robotic system for minimally invasive surgery," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1213–1220, 2013.

[28] T. D. Wortman, K. W. Strabala, A. C. Lehman, S. M. Farritor, and D. Oleynikov, "Laparoendoscopic singlesite surgery using a multifunctional miniature in vivo robot," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 7, no. 1, pp. 17–21, 2011.

[29] T. D. Wortman, A. Meyer, O. Dolghi, A. C. Lehman, R. L. McCormick, S. M. Farritor, and D. Oleynikov, "Miniature surgical robot for laparoendoscopic single-incision colectomy," *Surgical Endoscopy and Other Interventional Techniques*, vol. 26, no. 3, pp. 727–731, 2012.

[30] S. M. Lucas and C. P. Sundaram, "The MIMIC virtual reality trainer: stepping into three-dimensional, binocular, robotic simulation," in *Simulation Training in Laparoscopy and Robotic Surgery*. Springer, 2012, pp. 49–57.

[31] P. A. Kenney, M. F. Wszolek, J. J. Gould, J. A. Libertino, and A. Moinzadeh, "Face, content, and construct validity of dV-Trainer, a novel virtual reality simulator for robotic surgery," *Urology*, vol. 73, no. 6, pp. 1288–1292, 2009.

[32] "Mimic Simulation — dV-Trainer." [Online]. Available: http://www.mimicsimulation.com/products/dv-trainer/

[33] S. J. Phee, S. C. Low, V. A. Huynh, A. P. Kencana, Z. L. Sun, and K. Yang, "Master and slave transluminal endoscopic robot (MASTER) for natural orifice transluminal endoscopic surgery (NOTES)," *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society: Engineering the Future of Biomedicine, EMBC 2009*, pp. 1192–1195, 2009.

[34] B. Hannaford, J. Rosen, D. W. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. Ma, S. N. Kosari, and L. White, "Raven-II: An open platform for surgical robotics research," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 4, pp. 954–959, 2013.

[35] J. Rosen and B. Hannaford, "Doc at a distance," *IEEE Spectrum*, vol. 43, no. 10, pp. 34–39, 2006.

[36] A. Talasaz, “Haptics-enabled teleoperation for robotics- assisted minimally invasive surgery,” no. May, 2012.

[37] L. Dipietro, A. M. Sabatini, and P. Dario, “A survey of glove-based systems and their applications,” *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 38, no. 4, pp. 461–482, 2008.

[38] D. J. Sturman and D. Zeltzer, “A survey of glove-based input,” *Computer Graphics and Applications, IEEE*, vol. 14, no. 1, pp. 30–39, 1994.

[39] J. LaViola, “A survey of hand posture and gesture recognition techniques and technology,” *Brown University, Providence, RI*, 1999.

[40] G. Grimes, “Digital data entry glove interface device,” 1983.

[41] T. G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson, and Y. Harvill, “A hand gesture interface device,” in *ACM SIGCHI Bulletin*, vol. 18, no. 4. ACM, 1987, pp. 189–192.

[42] J. Hong and X. Tan, “Calibrating a VPL dataglove for teleoperating the Utah/MIT hand,” *1989 IEEE International Conference on Robotics and Automation*, pp. 1752–1757, 1989.

[43] S. C. Jacobsen, E. K. Iversen, D. F. Knutti, R. T. Johnson, and K. B. Biggers, “Design of the Utah M.I.T. dextrous hand,” *International Conference on Robotics and Automation*, pp. 1520–1532, 1986.

[44] L. Pao and T. Speeter, “Transformation of human hand positions for robotic hand control,” *Proceedings, 1989 International Conference on Robotics and Automation*, no. 1, pp. 1758–1763, 1989.

[45] S. Iba, J. Weghe, C. Paredis, and P. Khosla, “An architecture for gesture-based control of mobile robots,” *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, pp. 851–857, 1999.

[46] M. a. Diftler, “Evolution of the NASA / DARPA Robonaut control system,” *Control*, pp. 2543–2548, 2003.

[47] “Data Gloves - 5DT.” [Online]. Available: http://www.5dt.com/data-gloves/

[48] J. L. Hernandez-Rebollar, N. Kyriakopoulos, and R. W. Lindeman, “The AcceleGlove: a whole-hand input device for virtual reality,” *ACM SIGGRAPH 2002 conference abstracts and applications on - SIGGRAPH ’02*, p. 259, 2002.

[49] T. L. Baldi, M. Mohammadi, S. Scheggi, and D. Prattichizzo, “Using inertial and magnetic sensors for hand tracking and rendering in wearable haptics,” *IEEE World Haptics Conference, WHC 2015*, pp. 381–387, 2015.

[50] M. Cortese, M. Cempini, P. R. De Almeida Ribeiro, S. R. Soekadar, M. C. Carrozza, and N. Vitiello, “A mechatronic system for robot-mediated hand telerehabilitation,” *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 4, pp. 1753–1764, 2015.

[51] D. L. Quam, G. B. Williams, J. R. Agnew, and P. C. Browne, "An experimental determination of human hand accuracy with a dataglove," in *Proceedings of the Human Factors Society Annual Meeting*, vol. 33, no. 5. SAGE Publications Sage CA: Los Angeles, CA, 1989, pp. 315–319.

[52] N. X. Tran, H. Phan, V. V. Dinh, J. Ellen, B. Berg, J. Lum, E. Alcantara, M. Bruch, M. G. Ceruti, C. Kao, D. Garcia, S. Fugate, and L. Duffy, "Wireless data glove for gesture-based robotic control," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5611 LNCS, no. PART 2, pp. 271–280, 2009.

[53] J.-H. Kim, N. D. Thang, and T.-S. Kim, "3-D hand motion tracking and gesture recognition using a data glove," *2009 IEEE International Symposium on Industrial Electronics*, no. ISlE, pp. 1013–1018, 2009.

[54] P. Kumar, J. Verma, and S. Prasad, "Hand data glove: a wearable real-time device for human-computer interaction," *International Journal of Advanced Science and Technology*, vol. 43, pp. 15–26, 2012.

[55] R. N. Rohling and J. M. Hollerbach, "Calibrating the human hand for haptic interfaces," *Presence: Teleoperators and Virtual Environments*, vol. 2, no. 4, pp. 281–296, jan 1993.

[56] T. H. Speeter, "Transforming human hand motion for telemanipulation," *Presence: Teleoperators and Virtual Environments*, vol. 1, no. 1, pp. 63–79, jan 1992.

[57] B. A. Marcus, P. J. Churchill, and A. D. Little, "Sensing human hand motions for controlling dexterous robots," 1988.

[58] J. Makower, M. Parnianpour, and M. Nordin, "The validity assessment of the Dextrous Hand Master: a linkage system for the measurement of the joints of the hand," in *First World Congress of Biomechanics*, vol. 2.

[59] A. Wright and M. M. Stanisic, "Kinematic mapping between the EXOS Handmaster exoskeleton and the Utah/MIT dextrous hand," *IEEE International Conference on Systems Engineering*, pp. 101–104, 1990.

[60] M. Bouzit, G. Burdea, G. Popescu, and R. Boian, "The Rutgers Master II - new design force-feedback glove," *IEEE/ASME Transactions on Mechatronics*, vol. 7, no. 2, pp. 256–263, 2002.

[61] P. Olsson, S. Johansson, F. Nysjö, and I. Carlbom, "Rendering stiffness with a prototype haptic glove actuated by an integrated piezoelectric motor," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7282 LNCS, no. PART 1, pp. 361–372, 2012.

[62] S. Wise, W. Gardner, E. Sabelman, E. Valainis, Y. Wong, K. Glass, J. Drace, and J. M. Rosen, "Evaluation of a fiber optic glove for semi-automated goniometric measurements." *Journal of rehabilitation research and development*, vol. 27, no. 4, pp. 411–424, 1990.

[63] Y. Wu and T. S. Huang, "Vision-based gesture recognition: A review," in *International Gesture Workshop*. Springer, 1999, pp. 103–115.

[64] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 7, pp. 677–695, 1997.

[65] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding*, vol. 108, no. 1-2, pp. 52–73, 2007.

[66] Y. Wu and T. S. Huang, "Hand modeling, analysis and recognition," *IEEE Signal Processing Magazine*, vol. 18, no. 3, pp. 51–60, 2001.

[67] R. Watson, "A survey of gesture recognition techniques technical report tcd-cs-93-11," *Department of Computer Science, Trinity College Dublin*, vol. 2, p. 17, 1993.

[68] T. Starner and A. Pentland, "Real-time american sign language recognition from video using hidden markov models," in *Motion-Based Recognition.* Springer, 1997, pp. 227–243.

[69] J. Martin and J. Crowley, "An appearance-based approach to gesture-recognition," in *Image Analysis and Processing.* Springer, 1997, pp. 340–347.

[70] D. J. Sturman, "Whole-hand input," 1991.

[71] N. Shimada, K. Kimura, and Y. Shirai, "Real-time 3D hand posture estimation based on 2D appearance retrieval using monocular camera," in *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001. Proceedings. IEEE ICCV Workshop On.* IEEE, 2001, pp. 23–30.

[72] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using kinect sensor," *IEEE Transactions on Multimedia*, vol. 15, no. 5, pp. 1110–1120, 2013.

[73] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Efficient model-based 3D tracking of hand articulations using Kinect." in *BmVC*, vol. 1, no. 2, 2011, p. 3.

[74] M. Van Den Bergh, D. Carton, R. De Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz, D. Wollherr, L. Van Gool, and M. Buss, "Real-time 3D hand gesture interaction with a robot for understanding directions from humans," *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, pp. 357–362, 2011.

[75] M. G. Jacob, Y. T. Li, and J. P. Wachs, "A gesture driven robotic scrub nurse," *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, pp. 2039–2044, 2011.

[76] J. P. Wachs, H. I. Stern, Y. Edan, M. Gillam, J. Handler, C. Feied, and M. Smith, "A Gesture-based tool for sterile browsing of radiology images," *Journal of the American Medical Informatics Association*, vol. 15, no. 3, pp. 321–323, 2008.

[77] G. Ruppert, P. Amorim, T. F. Moraes, and J. Silva, "Touchless gesture user interface for 3D visualization using the Kinect platform and open-source frameworks," in *Innovative Developments in Virtual and Physical Prototyping - Proceedings of the 5th International Conference on Advanced Research and Rapid Prototyping*, 2012, pp. 215–219.

[78] K. O'Hara, N. Dastur, T. Carrell, G. Gonzalez, A. Sellen, G. Penney, A. Varnavas, H. Mentis, A. Criminisi, R. Corish, and M. Rouncefield, "Touchless interaction in surgery," *Communications of the ACM*, vol. 57, no. 1, pp. 70–77, 2014.

[79] Y. Kim, S. Leonard, A. Shademan, A. Krieger, and P. C. W. Kim, "Kinect technology for hand tracking control of surgical robots: Technical and surgical skill comparison to current robotic masters," *Surgical Endoscopy and Other Interventional Techniques*, vol. 28, no. 6, pp. 1993–2000, 2014.

[80] F. Weichert, D. Bachmann, B. Rudak, and D. Fisseler, "Analysis of the accuracy and robustness of the leap motion controller," *Sensors*, vol. 13, no. 5, pp. 6380–6393, 2013.

[81] D. Bassily, C. Georgoulas, J. Güttler, T. Linner, T. Bock, and T. U. München, "Intuitive and adaptive robotic arm manipulation using the leap motion controller," *Isr Robotik*, pp. 78–84, 2014.

[82] I. Staretu and C. Moldovan, "Leap Motion device used to control a real anthropomorphic gripper," *International Journal of Advanced Robotic Systems*, p. 1, 2016.

[83] I. Zubrycki and G. Granosik, "Using integrated vision systems: Three Gears and Leap Motion, to control a 3-finger dexterous gripper," in *Recent Advances in Automation, Robotics and Measuring Techniques SE - 52*, ser. Advances in Intelligent Systems and Computing, R. Szewczyk, C. Zieliński, and M. Kaliczyńska, Eds. Springer International Publishing, 2014, vol. 267, pp. 553–564.

[84] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. Dimaio, "An open-source research kit for the da Vinci Surgical System," in *IEEE International Conference on Robotics and Automation*, 2014, pp. 6434–6439.

[85] A. Deguet, R. Kumar, R. Taylor, and P. Kazanzides, "The cisst libraries for computer assisted intervention systems," *MIDAS J Syst Archit Comput Assist Interv*, pp. 1–8, 2008.

[86] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Mg, "ROS: an open-source robot operating system," *Icra*, vol. 3, p. 5, 2009.

[87] A. Colgan, "How does the leap motion controller work?" [Online]. Available: http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/

[88] Y. Kim, P. C. W. Kim, R. Selle, A. Shademan, and A. Krieger, "Experimental evaluation of contact-less hand tracking systems for tele-operation of surgical tasks," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2014, pp. 3502–3509. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6907364

[89] D. Bachmann, F. Weichert, and G. Rinkenauer, "Evaluation of the leap motion controller as a new contact-free pointing device," *Sensors (Switzerland)*, vol. 15, no. 1, pp. 214–233, 2014.

[90] H. Jin, Q. Chen, Z. Chen, Y. Hu, and J. Zhang, "Multi-LeapMotion sensor based demonstration for robotic refine tabletop object manipulation task," *CAAI Transactions on Intelligence Technology*, vol. 1, pp. 104–113, 2016.

[91] "Aurora." [Online]. Available: https://www.ndigital.com/medical/products/aurora/

[92] Northern Digital, *Aurora tracking system user manual*, 2010.

[93] G. F. Buess, A. Arezzo, M. O. Schurr, F. Ulmer, H. Fisher, L. Gumb, T. Testa, and C. Nobman, "A new remote-controlled endoscope positioning system for endoscopic solo surgery: The FIPS endoarm," *Surgical Endoscopy*, vol. 14, no. 4, pp. 395–399, 2000.

[94] S. Eslamian, L. A. Reisner, B. W. King, and A. K. Pandya, "Towards the implementation of an autonomous camera algorithm on the da Vinci platform." *Medicine Meets Virtual Reality 22: NextMed/MMVR22*, vol. 220, p. 118, 2016.

[95] L. Mettler, M. Ibrahim, and W. Jonat, "One year of experience working with the aid of a robotic assistant (the voice-controlled optic holder AESOP) in gynaecological endoscopic surgery." *Human Reproduction*, vol. 13, no. 10, pp. 2748–2750, 1998.

[96] B. M. Kraft, C. Jäger, K. Kraft, B. J. Leibl, and R. Bittner, "The AESOP robot system in laparoscopic surgery: Increased risk or advantage for surgeon and patient?" *Surgical Endoscopy And Other Interventional Techniques*, vol. 18, no. 8, pp. 1216–1223, 2004.

[97] A. Pandya, L. Reisner, B. King, N. Lucas, A. Composto, M. Klein, and R. Ellis, "A review of camera viewpoint automation in robotic and laparoscopic surgery," *Robotics*, vol. 3, no. 3, pp. 310–329, 2014.

[98] J. Gilbert, "The EndoAssist robotic camera holder as an aid to the introduction of laparoscopic colorectal surgery," *Ann R Coll Surg Engl*, vol. 91, pp. 389–393, 2009.

[99] G.-Q. Wei, K. Arbter, and G. Hirzinger, "Real-time visual servoing for laparoscopic surgery. Controlling robot motion with color image segmentation," *IEEE Engineering in Medicine and Biology Magazine*, vol. 16, no. 1, pp. 40–45, 1997.

[100] K. Omote, H. Feussner, A. Ungeheuer, K. Arbter, G.-Q. Wei, J. Siewert, and G. Hirzinger, "Self-guided robotic camera control for laparoscopic surgery compared with human camera control," *The American Journal of Surgery*, vol. 177, no. 4, pp. 321–324, 1999.

[101] G. F. Buess, A. Arezzo, M. O. Schurr, F. Ulmer, H. Fisher, L. Gumb, T. Testa, and C. Nobman, "A new remote-controlled endoscope positioning system for endoscopic solo surgery," *Surgical Endoscopy*, vol. 14, no. 4, pp. 395–399, 2000.

[102] B. King, L. Reisner, A. Pandya, A. Composto, R. Ellis, and M. Klein, "Towards an autonomous robot for camera control during laparoscopic surgery," *Journal of Laparoendoscopic & Advanced Surgical Techniques*, vol. 23, no. 12, pp. 1027–1030, 2013. [Online]. Available: http://online.liebertpub.com/doi/abs/10.1089/lap.2013.0304

[103] S. M. Ali, L. A. Reisner, B. King, A. Cao, G. Auner, M. Klein, and A. K. Pandya, "Eye gaze tracking for endoscopic camera positioning: an application of a hardware/software interface developed to automate Aesop." *Studies in health technology and informatics*, vol. 132, pp. 4–7, 2007.

[104] E. N. Arcoverde, R. M. Duarte, R. M. Barreto, and J. Paulo, "Enhanced real-time head pose estimation system for mobile device," vol. 21, no. January, pp. 281–293, 2014.

[105] S. O. H. Madgwick, A. J. L. Harrison, and R. Vaidyanathan, "Estimation of IMU and MARG orientation using a gradient descent algorithm," in *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on.* IEEE, 2011, pp. 1–7.

[106] J. L. Marins, X. Yun, E. R. Bachmann, R. B. McGhee, and M. J. Zyda, "An extended Kalman filter for quaternion-based orientation estimation using MARG sensors," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 4. IEEE, 2001, pp. 2003–2011.

[107] H. Ren and P. Kazanzides, "Investigation of attitude tracking using an integrated inertial and magnetic navigation system for hand-held surgical instruments," *IEEE/ASME Transactions on Mechatronics*, vol. 17, no. 2, pp. 210–217, apr 2012.

[108] K. Karolczak and A. Klepaczko, "A stereoscopic viewer of the results of vessel segmentation in 3D magnetic resonance angiography images." [Online]. Available: http://www.eletel.p.lodz.pl/angiosim/downloads-pdfs/karolczak{_}SPA2014.pdf

[109] N. A. Dodgson, "Variation and extrema of human interpupillary distance," in *Electronic imaging 2004.* International Society for Optics and Photonics, 2004, pp. 36–46.

[110] T. Kot and P. Novák, "Utilization of the Oculus Rift HMD in mobile robot teleoperation," *Applied Mechanics and Materials*, vol. 555, pp. 199–208, 2014.

[111] D. Mintz, V. Falk, and J. Salisbury, "Comparison of three high-end endoscopic visualization systems on telesurgical performance," in *Medical Image Computing and Computer-Assisted Intervention Äì MICCAI 2000*, 2000, vol. 1935, pp. 57–119. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-40899-4{_}39

[112] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti, "Surround-screen projection-based virtual reality: the design and implementation of the CAVE," in *Proceedings of the 20th annual conference on Computer graphics and interactive techniques.* ACM, 1993, pp. 135–142.

[113] A. Davis, "How to build your own leap motion art installation," 2014. [Online]. Available: http://blog.leapmotion.com/how-to-build-your-own-leap-motion-art-installation/

Appendix A

Permissions and Approvals

The following forms and permission statements are presented in this Appendix.

- Ethics approval for the experimental evaluation from the Research Ethics Board for Health Sciences at Western University
- Ethics approval for the experimental evaluation from Lawson Health Research Institute
- Online permission from Springer for Figure 2.5
- Written permission from Cyber Glove Systems for Figures 2.8 and 2.10
- Written permission from Leap Motion, Inc. for Figures 2.15, 2.16, 2.17, and 2.18
- Written permission from TransEnterix for Figures 2.4a and 2.4b
- Written permission from Titan Medical, Inc. for Figure 2.2a
- Written permission from IOS Press for Figure 4.4

Western Research

Research Ethics

Western University Health Science Research Ethics Board
HSREB Delegated Initial Approval Notice

Principal Investigator: Prof. Rajnikant Patel
Department & Institution: Engineering\Electrical & Computer Engineering, Western University

Review Type: Delegated
HSREB File Number: 109243
Study Title: Evaluation of a Noncontact method of control for Robotics-Assisted Surgery

HSREB Initial Approval Date: May 29, 2017
HSREB Expiry Date: May 29, 2018

Documents Approved and/or Received for Information:

Document Name	Comments	Version Date
Western University Protocol	Received May 29, 2017	
Recruitment Items	Email Script - Received April 10, 2017	
Letter of Information & Consent		2017/05/29
Instruments	Questionnaire - Received April 10, 2017	
Instruments	Standard Instruments - Received May 13, 2017	

The Western University Health Science Research Ethics Board (HSREB) has reviewed and approved the above named study, as of the HSREB Initial Approval Date noted above.

HSREB approval for this study remains valid until the HSREB Expiry Date noted above, conditional to timely submission and acceptance of HSREB Continuing Ethics Review.

The Western University HSREB operates in compliance with the Tri-Council Policy Statement Ethical Conduct for Research Involving Humans (TCPS2), the International Conference on Harmonization of Technical Requirements for Registration of Pharmaceuticals for Human Use Guideline for Good Clinical Practice Practices (ICH E6 R1), the Ontario Personal Health Information Protection Act (PHIPA, 2004), Part 4 of the Natural Health Product Regulations, Health Canada Medical Device Regulations and Part C, Division 5, of the Food and Drug Regulations of Health Canada.

Members of the HSREB who are named as Investigators in research studies do not participate in discussions related to, nor vote on such studies when they are presented to the REB.

The HSREB is registered with the U.S. Department of Health & Human Services under the IRB registration number IRB 00000940.

LAWSON
HEALTH RESEARCH INSTITUTE

LAWSON FINAL APPROVAL NOTICE

LAWSON APPROVAL NUMBER: R-17-205

PROJECT TITLE: Evaluation of a Noncontact method of control for Robotics-Assisted Surgery

PRINCIPAL INVESTIGATOR:	Dr. Rajnikant Patel
LAWSON APPROVAL DATE:	June 1, 2017
Health Sciences REB#:	109243

Please be advised that the above project was reviewed by the Clinical Research Impact Committee and Lawson Administration and the project:

Was Approved

Please provide your Lawson Approval Number (R#) to the appropriate contact(s) in supporting departments (eg. Lab Services, Diagnostic Imaging, etc.) to inform them that your study is starting. The Lawson Approval Number must be provided each time services are requested.

Dr. David Hill
V.P. Research
Lawson Health Research Institute

All future correspondence concerning this study should include the Lawson Approval Number and should be directed to Sherry Paiva, Research Approval Officer, Lawson Health Research Institute, 750 Baseline Road, East, Suite 300.

cc: Administration

SPRINGER LICENSE
TERMS AND CONDITIONS

Jun 02, 2017

This Agreement between Western University -- Cameron Dawson ("You") and Springer ("Springer") consists of your license details and the terms and conditions provided by Springer and Copyright Clearance Center.

License Number	4117950603720
License date	May 28, 2017
Licensed Content Publisher	Springer
Licensed Content Publication	Surgical Endoscopy
Licensed Content Title	Miniature surgical robot for laparoendoscopic single-incision colectomy
Licensed Content Author	Tyler D. Wortman
Licensed Content Date	Jan 1, 2011
Licensed Content Volume	26
Licensed Content Issue	3
Type of Use	Thesis/Dissertation
Portion	Figures/tables/illustrations
Number of figures/tables/illustrations	1
Author of this Springer article	No
Order reference number	
Original figure numbers	figure 2
Title of your thesis / dissertation	Design and Evaluation of a Non-Contact Interface for Minimally Invasive Robotic Surgery
Expected completion date	Jun 2017
Estimated size(pages)	120
Requestor Location	Western University 1151 Richmond St

From: Faisal Yazadi
Sent: Monday, May 29, 2017 7:16 PM
To: Cameron James Dawson
Subject: CyberGlove

Dear Mr. Dawson,

Thank you for choosing CyberGlove products for your thesis and publication.
You are welcome to use our product images.

Let me know if any questions.

Regards
Faisal

Faisal Yazadi
CyberGlove Systems LLC

From: Tom Kaweski |
Sent: Monday, May 22, 2017 7:04 PM
To: Cameron James Dawson
Subject: Fwd: [Research / Education] Western University

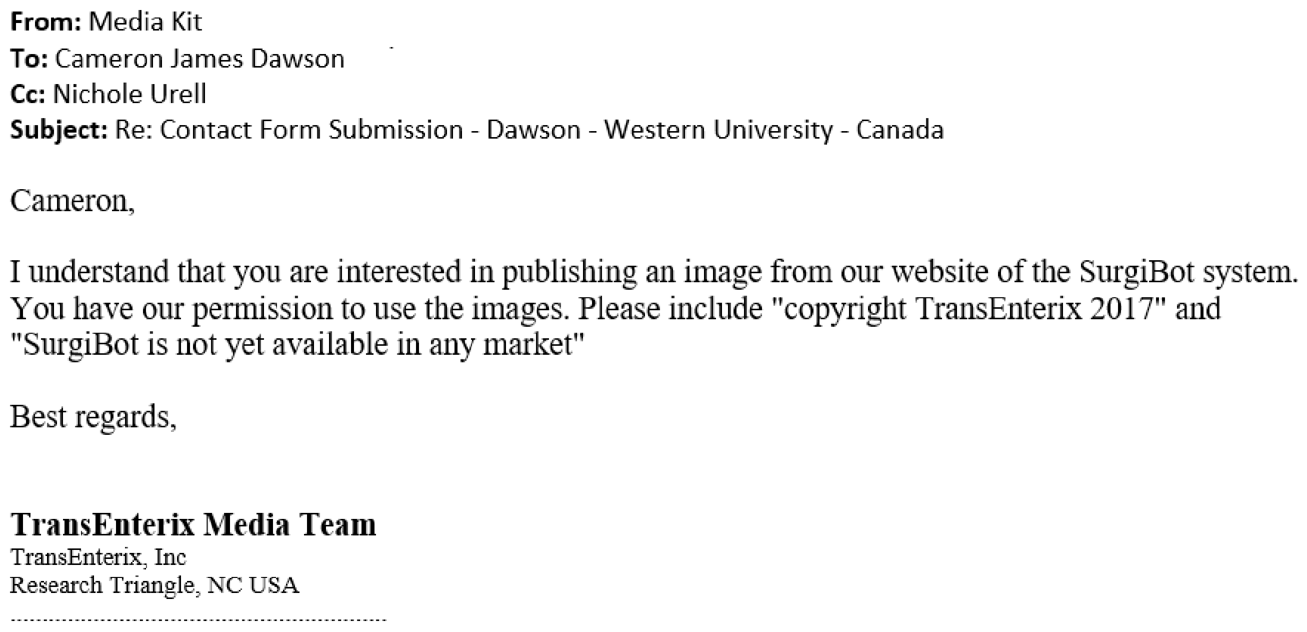
Hi Cameron,

Thanks for your interest in Leap Motion.

The copyright in all the images you reference is owned by Leap Motion. We hereby provide permission for you to use the images in your thesis, with attribution.

Regards,
Tom Kaweski

Tom Kaweski | General Counsel |
www.leapmotion.com



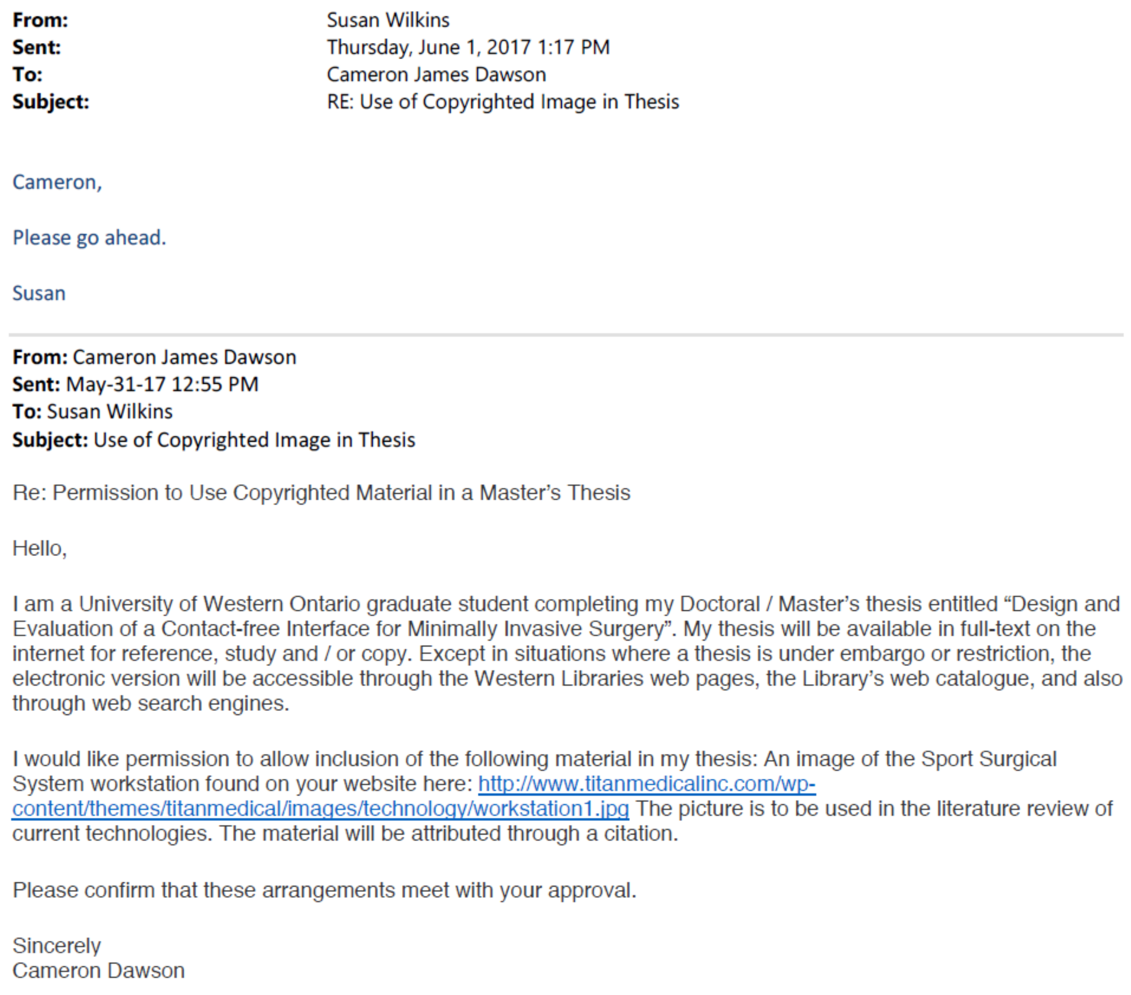
From: Media Kit
To: Cameron James Dawson
Cc: Nichole Urell
Subject: Re: Contact Form Submission - Dawson - Western University - Canada

Cameron,

I understand that you are interested in publishing an image from our website of the SurgiBot system. You have our permission to use the images. Please include "copyright TransEnterix 2017" and "SurgiBot is not yet available in any market"

Best regards,

TransEnterix Media Team
TransEnterix, Inc
Research Triangle, NC USA



From: Susan Wilkins
Sent: Thursday, June 1, 2017 1:17 PM
To: Cameron James Dawson
Subject: RE: Use of Copyrighted Image in Thesis

Cameron,

Please go ahead.

Susan

From: Cameron James Dawson
Sent: May-31-17 12:55 PM
To: Susan Wilkins
Subject: Use of Copyrighted Image in Thesis

Re: Permission to Use Copyrighted Material in a Master's Thesis

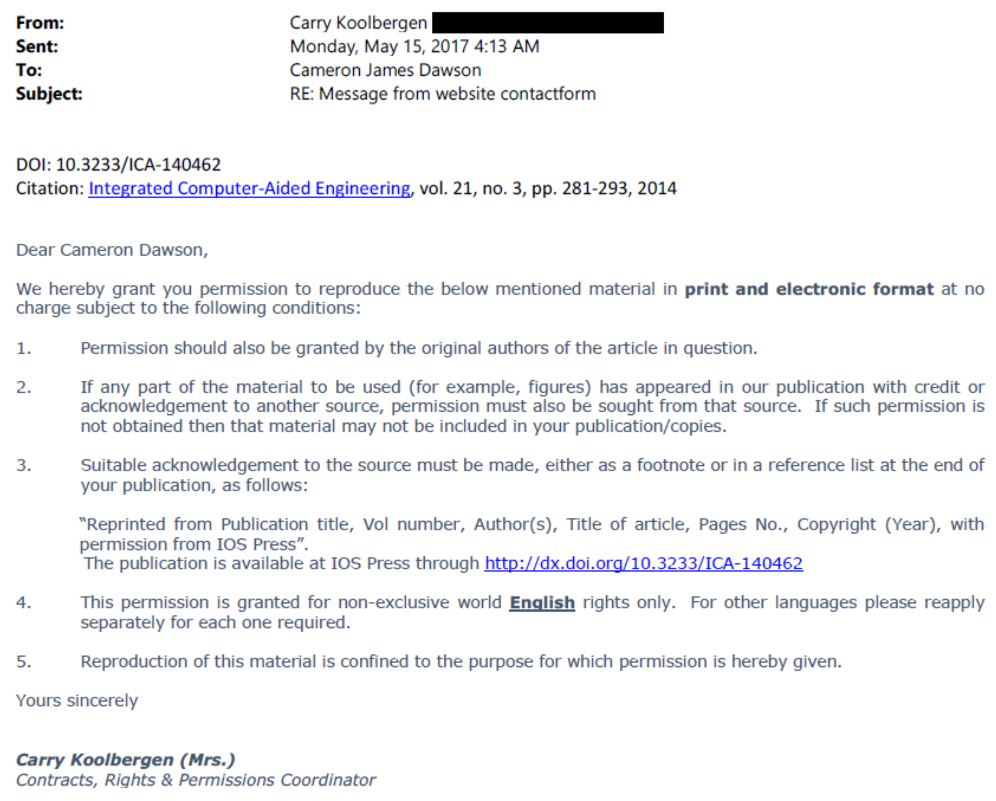
Hello,

I am a University of Western Ontario graduate student completing my Doctoral / Master's thesis entitled "Design and Evaluation of a Contact-free Interface for Minimally Invasive Surgery". My thesis will be available in full-text on the internet for reference, study and / or copy. Except in situations where a thesis is under embargo or restriction, the electronic version will be accessible through the Western Libraries web pages, the Library's web catalogue, and also through web search engines.

I would like permission to allow inclusion of the following material in my thesis: An image of the Sport Surgical System workstation found on your website here: http://www.titanmedicalinc.com/wp-content/themes/titanmedical/images/technology/workstation1.jpg The picture is to be used in the literature review of current technologies. The material will be attributed through a citation.

Please confirm that these arrangements meet with your approval.

Sincerely
Cameron Dawson



From: Carry Koolbergen [redacted]
Sent: Monday, May 15, 2017 4:13 AM
To: Cameron James Dawson
Subject: RE: Message from website contactform

DOI: 10.3233/ICA-140462
Citation: Integrated Computer-Aided Engineering, vol. 21, no. 3, pp. 281-293, 2014

Dear Cameron Dawson,

We hereby grant you permission to reproduce the below mentioned material in **print and electronic format** at no charge subject to the following conditions:

1. Permission should also be granted by the original authors of the article in question.

2. If any part of the material to be used (for example, figures) has appeared in our publication with credit or acknowledgement to another source, permission must also be sought from that source. If such permission is not obtained then that material may not be included in your publication/copies.

3. Suitable acknowledgement to the source must be made, either as a footnote or in a reference list at the end of your publication, as follows:

   "Reprinted from Publication title, Vol number, Author(s), Title of article, Pages No., Copyright (Year), with permission from IOS Press".
   The publication is available at IOS Press through http://dx.doi.org/10.3233/ICA-140462

4. This permission is granted for non-exclusive world **English** rights only. For other languages please reapply separately for each one required.

5. Reproduction of this material is confined to the purpose for which permission is hereby given.

Yours sincerely

Carry Koolbergen (Mrs.)
Contracts, Rights & Permissions Coordinator

VITA

Name:	Cameron J. Dawson
Post-secondary Education and Degrees:	The University of Western Ontario London, Ontario, Canada 2011–2015 B.E.Sc., Mechatronic Systems Engineering
Honours and Awards:	Western Graduate Research Scholarship
Related Work Experience:	Teaching Assistant *MSE 2201 – Introduction to Electrical Instrumentation* *MSE 3302 – Sensors and Actuators* *MME 2259 – Product Design and Development* The University of Western Ontario 2015–2017
	Research Assistant The University of Western Ontario 2015–2017