Complementary Situational Awareness for an Intelligent Telerobotic Surgical Assistant System

by

Preetham Chalasani

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

October, 2018

Abstract

Robotic surgical systems have contributed greatly to the advancement of Minimally Invasive Surgeries (MIS). More specifically, telesurgical robots have provided enhanced dexterity to surgeons performing MIS procedures. However, current robotic teleoperated systems have only limited situational awareness of the patient anatomy and surgical environment that would typically be available to a surgeon in an open surgery. Although the endoscopic view enhances the visualization of the anatomy, perceptual understanding of the environment and anatomy is still lacking due to the absence of sensory feedback.

In this work, these limitations are addressed by developing a computational framework to provide Complementary Situational Awareness (CSA) in a surgical assistant. This framework aims at improving the human-robot relationship by providing elaborate guidance and sensory feedback capabilities for the surgeon in complex MIS procedures. Unlike traditional teleoperation, this framework enables the user to telemanipulate the situational model in a virtual environment and uses that information to command the slave robot with appropriate admittance gains and environmental

constraints. Simultaneously, the situational model is updated based on interaction of the slave robot with the task space environment.

However, developing such a system to provide real-time situational awareness requires that many technical challenges be met. To estimate intraoperative organ information continuous palpation primitives are required. Intraoperative surface information needs to be estimated in real-time while the organ is being palpated/scanned. The model of the task environment needs to be updated in near real-time using the estimated organ geometry so that the force-feedback applied on the surgeon's hand would correspond to the actual location of the model. This work presents a real-time framework that meets these requirements/challenges to provide situational awareness of the environment in the task space. Further, visual feedback is also provided for the surgeon/developer to view the near video frame rate updates of the task model. All these functions are executed in parallel and need to have a synchronized data exchange. The system is very portable and can be incorporated to any existing telerobotic platforms with minimal overhead.

ABSTRACT

Thesis Committee

Russell H.Taylor (Advisor)
John C. Malone Professor
Department of Computer Science
The Johns Hopkins University

Peter Kazanzides
Research Professor
Department of Computer Science
The Johns Hopkins University

Marin Kobilarov
Assistant Professor
Department of Mechanical Engineering
The Johns Hopkins University

Acknowledgments

During the course of my doctoral studies, I have come across many people and have had numerous memorable experiences. This thesis wouldn't have been possible without them, and to whom I am greatly indebted.

First and foremost, I would like to express my appreciation and deepest gratitude to my advisor, Dr. Russel H. Taylor, for his steadfast support, guidance, and encouragement throughout my time as his student. I have been extremely lucky to have an advisor who cared so much about my work, and who responded to my questions and queries so promptly. His creativity, enthusiasm, and dedication towards his students' research is unparalleled. Despite his extremely busy schedule, he was always available for brainstorming sessions with us. Thanks to Dr. Taylor and his mentorship, I have had a wonderfully fulfilling Ph.D. experience and wouldn't have it any other way.

In addition to my advisor, I would like to sincerely thank members of my thesis committee, Dr. Peter Kazanzides and Dr. Marin Kobilarov for their added guidance, careful attention to detail; and insightful suggestions to strengthen this manuscript. Dr. Kazanzides has helped me improve my software engineering skills in various

aspects of my research and has provided technical guidance throughout my time in the laboratory. Dr. Kobilarov had also helped me publish my first conference paper, which went on to be an integral part of my thesis. In addition, I am also thankful to Dr. Kazanzides and Dr. Kobilarov, for participating in my GBO.

I would like to acknowledge the support I received from National Science Foundation, which provided the funding for this work through NSF NRI grant IIS-1327657 and partly from Johns Hopkins University internal funds. I am grateful to have worked with Dr. Nabil Simaan and his team (Long Wang, Rashid Yasin, Colette Abah and Rajarshi Roy) from Vanderbilt University, and Dr. Howie Choset and his team (Elif Ayvali, Arun Srivatsan Rangaprasad, and Nicolas Zevallos) from Carnegie Mellon University. I am also extremely grateful to Anton Deguet for his guidance and support, not only on this grant but throughout my time as a doctoral student at Johns Hopkins University. His technical expertise and relentless help were instrumental for the completion of this work.

Furthermore, I am immensely thankful to various professors for their invaluable advice during the course of my research: Dr. Scott Smith and Dr. Louis L. Whitcomb, for having been part of my GBO committee; Dr. Misha Kazhdan, for his guidance during my first year at Hopkins; Dr. Greg Hager, for providing me with an opportunity to work on the Balaur Wall project that greatly helped me improve my coding skills; Dr. Iulian Iordachita, for his help in discussing mechanical design and manufacturing problems, and for letting me to use his lab equipment; and lastly,

ACKNOWLEDGMENTS

Dr. Emad Boctor, for his help and expertise in integrating the ultrasound sensing modality into the CSA framework.

A special thanks to many current and former colleagues whose vast expertise has made my Ph.D. journey pleasant and relatively easy. Thank you, Marcin Balicki, Balazs Vagvolgyi, Kelleher Guerin and Kevin Olds, for mentoring me on various projects that complemented my research. Additionally, I am much obliged to all the members of LCSR, who have made my time at Hopkins a fun and memorable experience - Zihan Chen, Berk Gonec, Farshid Alambeigi, Rob Grupp, Ayushi Sinha, Ali Ebrahimi, Purnima Rajan, Amirhossein Farvardin, Jonathan Bohren, Mohit Singhala, Paul Wilkening, Seth Billings, Baichuan Jiang, Zhaoshuo Li, Niravkumar Patel, Xingchi He, Ehsan Azimi, and Simon Leonard. I would like to especially thank Mahya Shahbazi for helping me proofread this manuscript.

Lastly, my gratitude goes to the graduate program for providing me the best education that one could possibly have at the Johns Hopkins University. I would also like to thank the exceptional staff of the Department of Computer Science and LCSR. Thank you, Zack Burwell, Lorrie Dodd, Ashley Moriarty, Jordan Card, Alison Morrow, Julia Ortiz-Foy, Laura Graham, Tracy Marshall, Cathy Thornton, Debbie DeFord, and Jamie Meehan, for helping me with the administrative aspects of my Ph.D. program.

To all my friends outside my academic community - Kartik, Rishabh, Berry, Gaurav, Nivi, Lucky, Nairuthya, Sid, and Vignesh, thank you for listening and supporting

me through this entire process. I will forever remember the late night food hunts, last-minute holiday trips, weekend game nights, occasional night-outs, and the unavoidable controversial arguments. Most importantly, a special thanks to my girlfriend, Swathi Karthikeyan, for standing by me since day one; and supporting me unconditionally throughout this entire process. Last but not least, I owe my deepest gratitude to my dearest brother, Dheeraj, for his boundless encouragement and support. Ever since I was a kid, he was, and always will be, my role model.

Finally, I thank my parents, Suneetha and Kutumbarao Chalasani, and my extended family, especially my grandparents, for always being supportive of my education. Although they are all back home in India, there has never been a time when I have felt alone, whether in the writing of this thesis or in the general attainment of my education. For that, I remain eternally grateful.

Dedication

"God loves you unconditionally, as you are and not as you should be, because nobody is as they should be." - Brennan Manning

(PS: The only gods I know and believe in, are my parents.)

Mom & Dad, thanks for everything.

This is for you.

Contents

List of Tables

LIST OF TABLES

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Background

Minimally Invasive Surgeries (MISs) are redefining the field of medicine. MIS includes both laparoscopy (surgery through small holes) and endoscopy (diagnostic and therapeutic procedures performed through the body's organs and vessels). In MIS, the surgeon makes small holes, usually less than half an inch rather than making a large incision. The surgeon then inserts specially designed thin instruments and sophisticated video equipment to perform the operation through that small opening, making the whole procedure minimally invasive. MIS is based on three basic requirements: availability of high-quality videoendoscopic images, use of high precision surgical instruments and high dexterity in the execution of surgical maneuvers, with personnel who have to be specifically trained to improve their surgical skills.

Currently, MIS is impacting all surgical specialties and can be divided into two levelsbasic and advanced. Basic laparoscopic skills are one-handed that involve simple organ removal, require limited vascular control and no reconstruction, and can be performed safely by almost any surgeon. Advanced MIS procedures require two-handed skills, such as bimanual manipulation, suturing, and knot tying. Depending on the procedure, MIS can be performed with the surgeon manipulating the instruments by hand or with robot-assistance. Hand-assisted laparoscopic surgery devices allow the surgeon to overcome skill and instrument limitations and this offers patients a portion of the benefits of MIS. However, the introduction of robotic technology can enhance human visualization, strength, precision, and degrees of motion in performing complex surgical tasks.[1]

In the mid-80's, the idea of robots in surgery was a proof of concept rather than an advantage over traditional surgery. Since then, significant developments have been made in the field of robotic surgery. Telerobotics is considered to be an integral part of the wider field of telemedicine. Robotic systems have made an impact in various medical disciplines including general surgery, neurosurgery and orthopedic surgery.[2–6] There have been multiple robot-assisted surgeries and developments that include cholecystectomy,[7] prostatectomy,[8] hip-replacement[9,10] and lower abdominal laparoscopy.[11] Some of the advantages of robot-assisted surgery include the elimination of fatigue for labor-intensive movements, increased precision, ease of use and motion scaling.

CHAPTER 1. INTRODUCTION

Teleoperation or telerobotics is defined as the operation of a machine or a robot at a distance. It is similar in meaning to the phrase "remote control" but is usually encountered in research, academic and technical environments. In telerobotic systems, the remote manipulator is controlled from the operator's site by sending position or velocity commands while receiving visual and other sensory feedback information. The local system operated by the user is called the "master" robot, and the remote system that is being manipulated is called the "slave" robot, and the overall system is referred to as a "master-slave system". This master-slave paradigm is used for many applications. A computer interface is used to aid in the planning, command, and control of the robots. In 1988, Machida *et al.*[12] demonstrated a system on which the supervisor could teach a computer-modeled telerobot and receive force-feedback on the master hand if mechanical interference occurs between the modeled slave hand and its environment. Park *et al.*[13](1991) demonstrated a computer-aided technique for commanding a telerobot to move to a goal point while avoiding objects. In 1992, Funda *et al.*[14] extended the work of Machida *et al.*, and the key feature in their work is that the instructions sent to the robot are generated automatically in a more compact form than record and playback of analog signals. The operator commands the robots by kinesthetic as well as visual interactions with a virtual computer simulation.

In the late 90s, the interest shifted to other areas such as space,[15–21] medicine,[22–27] undersea[28–31] or mobile[32–38] robotics. Efforts were accelerated by the availability of increasing computer power as well as the introduction of novel hand controllers, *e.g.*,

the PHANToM device.[39] With the first transatlantic telesurgery demonstration in 2001, Computer Motion demonstrated the feasibility of telerobotic systems even in the delicate field of surgery.[40] A surgeon in New York (USA) used a ZEUS[41] system to perform a laparoscopic cholecystectomy on a patient located in Strasbourg (France). The use of telerobotics in the field of medicine, in particular, MIS procedures, has been increasing. In a robotic MIS, the surgeon or doctor holds on to a master device and uses it to control the movement of the robotic instrument attached to a slave robot. The slave manipulator is programmed to track the master device. Many medical robotic systems employ teleoperation as the major mode of operation while both the master manipulator and the slave remote manipulator are located in the same room.

One key factor that enhances the performance of a telerobotic system is telepresence. Telepresence means that the information about the remote environment is presented to the surgeon/operator in a natural fashion, which implies a feeling of presence at the remote site. A good degree of telepresence guarantees the feasibility of the required manipulation task.[42] Telepresence in a telerobotic system is generally classified as *weak* or *strong*. If the operator gives symbolic commands by pushing buttons on the master and watches the resulting action in the remote environment, its coupling is rather weak. Some degree of "intelligence" is required for a remote robot to execute such symbolic commands. The coupling is comparably strong in a bilateral teleoperation scenario. Generally, the motion (and/or force) of the human

operator initiates the motion of the master robot. The motion of the master robot is transmitted to the slave robot that tracks the motion of the master robot. During the task execution, the interactions of the slave robot with the remote environment are sensed, communicated and fed back to the operator through a multi-modal human-system interface. Thus, to develop a real-time multi-modal teleoperation framework, one has to make sure to have strong control distribution between the operator and the slave robot controller.

Existing Telerobotic Research Platforms

Following are few telerobotic research platforms and slave arms that are commonly used in medical robotics:

1. **dVRK**[43] : da Vinci Research Kit (dVRK) is one of the most capable research platforms in surgical robotics developed from the first-generation da Vinci® Surgical System (Intuitive Surgical, Inc., Sunnyvale, CA). The da Vinci System can be configured (by the manufacturer) to have a read-only interface to both master and slave manipulators for research purposes, limiting access to the rest of the control architecture. The dVRK provides open-source control electronics, firmware, and software to control research systems based on the first generation da Vinci System. This enables researchers to modify control algorithms and test new control methods, including autonomous and semi-autonomous control.

   The software architecture of dVRK[44] consists of a distributed hardware interface

layer, a real-time component-based software framework, and integration with Robot Operating System (ROS). Figure 1.1 shows the dVRK setup at Johns Hopkins Universty with two Master Tool Manipulators (MTMs) and two Patient Side Manipulators (PSMs).

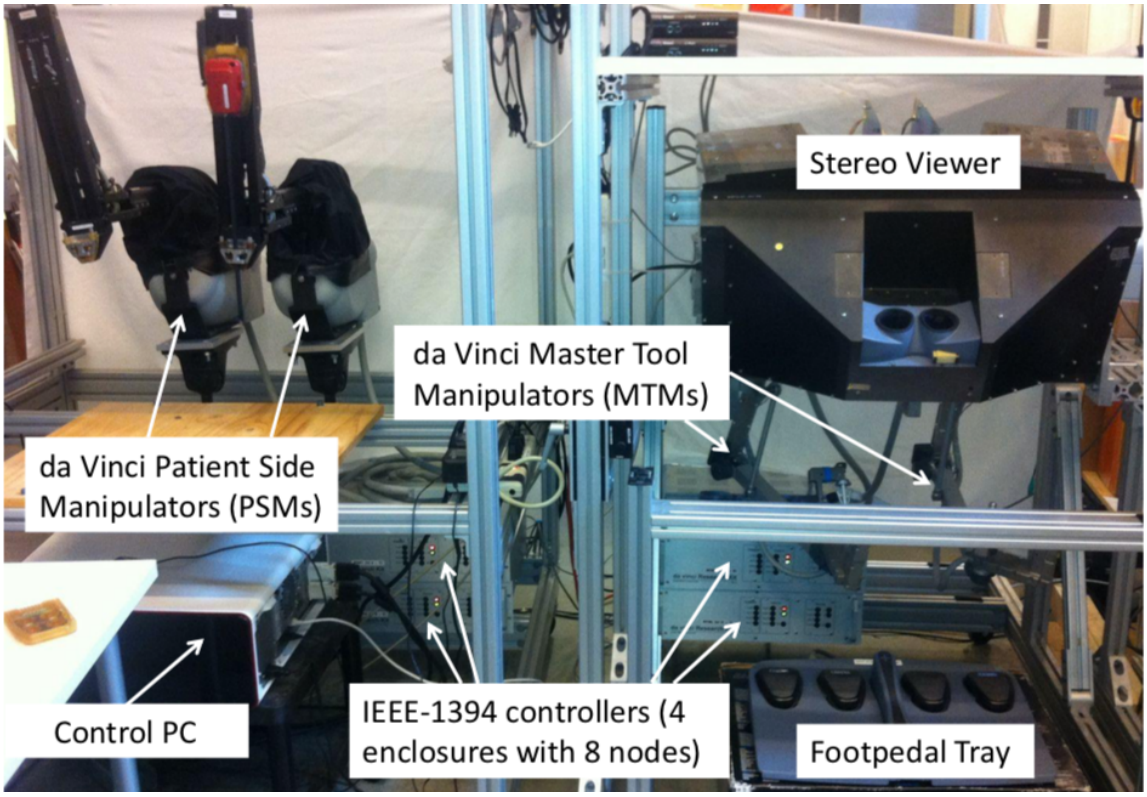


Figure 1.1: The dVRK Research Platform (two MTMs, two PSMs)

2. **Raven II**:[45] The Raven II is an open-architecture surgical robot for laparoscopic surgery research from Applied Dexterity. It has two 3-Degree of Freedom (DOF) spherical positioning mechanisms capable of attaching interchangeable 4-DOF instruments. This system is intended to facilitate collaborative research on advances in a surgical robot.

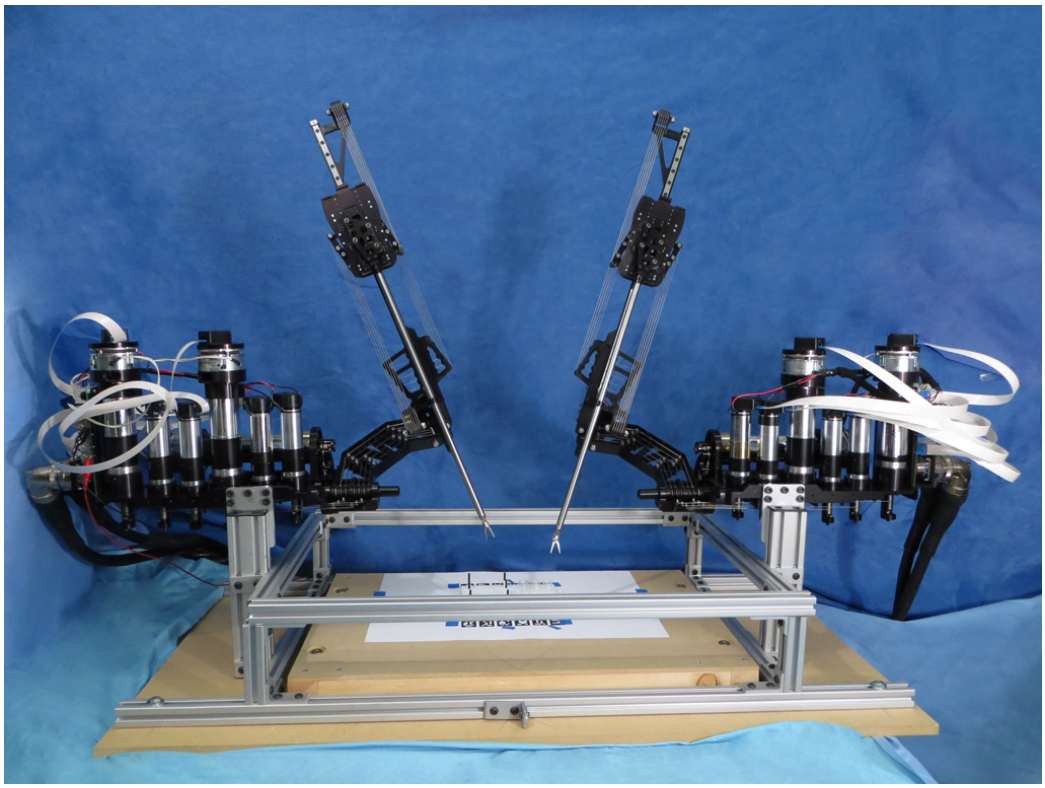


Figure 1.2: The Raven II surgical robot

3. **UR3**:[46] Universal Robots UR3 (Universal Robots, Odense, Denmark) is a collaborative table-top robot used for light assembly tasks and automated workbench scenarios. Applications of the UR3 robot span manufacturing industries

from medical devices to circuit boards and electronic components. The company provides documentation for communicating with the robot over a network using the Transmission Control Protocol (TCP). The research group working on dVRK at Johns Hopkins University have developed an open source software package using the cisst libraries to control the robot. This allows the research community to use Universal Robots (UR3, UR5 or UR10) as a slave robot or as a cooperatively-controlled robot.

Figure 1.3: The Universal Robots UR3

4. **LBR iiwa**[47] : LBR iiwa is a 7 DOF robot produced by a German multinational, KUKA. LBR stands for Leichtbauroboter (German for lightweight robot) and iiwa for intelligent industrial work assistant. It is mainly designed for interactions with humans and is therefore equipped with torque sensors after the gearbox of each actuated joint, which allows for cooperative interaction control.

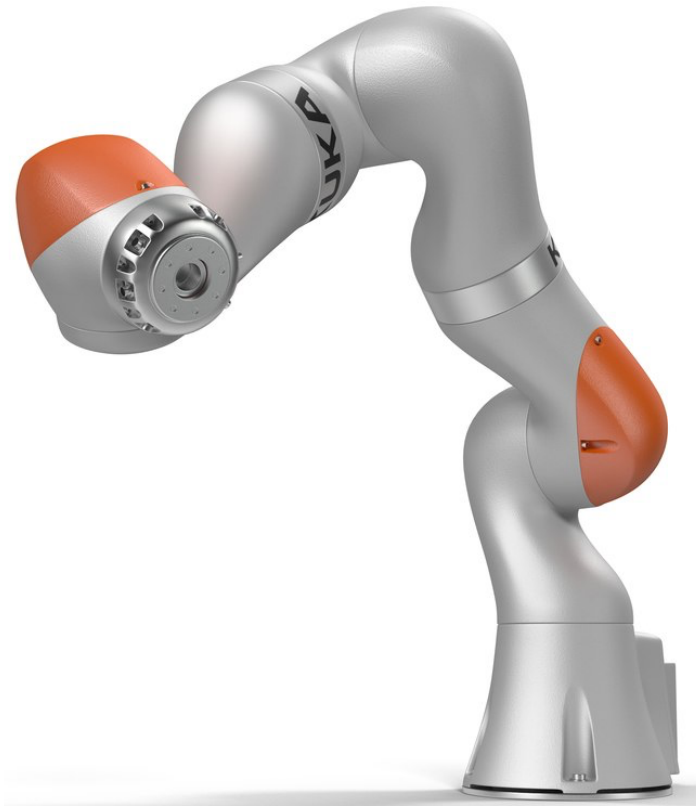


Figure 1.4: LBR iiwa by KUKA

Model Mediated Teleoperation (MMT)

Integration of model mediation into teleoperation frameworks is an effective approach to enable strong telepresence in such systems.[48] In the MMT approach, a model of the environment at the slave side is employed on the master side, approximating the task space/environment at the slave side. The model parameters describing an object in the slave environment are continuously estimated in real-time and transmitted back to the master robot whenever an updated model of the environment is obtained at the slave side. On the master side, the local virtual model of the environment is reconstructed or updated on the basis of the received model parameters, and the haptic feedback is computed on the basis of the local model. The Model Mediated Teleoperation (MMT) approach is usually used in bilateral telemanipulation systems under large communication delays ranging from hundreds of milliseconds to several seconds. One key application where MMT is useful in is on-orbit satellite servicing[49] where the goal is to repair, refurbish or refuel a satellite that has already been launched. In this case, for a servicing spacecraft to refuel the remote satellite, it is first needed to cut the tape that secures the patch of multi-layer insulation covering the satellite access panel. This task is challenging due to communication delay of several seconds. This task was modeled as sliding a tool along a planar surface, and the use of MMT allows the user to experience lag-free haptic feedback even though there is a large communication delay. While MMT is useful in bilateral telemanipulation under the delay, the paradigm can be also advantageous in the absence of commu-

nication delay. MMT enables development of a high-fidelity telerobotic framework where the model of the task environment can be updated in real-time without any communication delay in applications such as robot-assisted MIS. A more detailed background on MMT is discussed in Chapter 4.

The work discussed in this dissertation is focused on developing a strongly coupled multi-modal teleoperation system to provide Complementary Situational Awareness (CSA) in a surgical assistant. Unlike traditional telerobotic systems, this framework uses model-mediated teleoperation and monitors the interaction of the slave robot with the remote environment. The information is then used to update the model of the remote environment in real-time for teleoperation. The framework is developed in a modular fashion to readily incorporate CSA into any existing open-source robotic platform.

1.2 Relevant Prior Work

There is considerable prior art describing capabilities that can be implemented in an MMT to provide more information to the surgeon. Sotiras et al.[50] presented a review of various deformable registration methods dealing with environments that deform relative to *a priori* models. In more recent work,[51,52] global deformation of a model was addressed.

A number of researchers[53–58] have developed finger-like tactile and force sensors

to provide information about tool-to-tissue interaction forces. Mahvash et al.[59] developed a control system for the da Vinci surgical system,[60] which provides force feedback with a position-position controller with friction and inertia compensation. Using this system they provided some results on stiffness estimation of a tissue model, based on discrete palpation. More recently, Xu and Simaan have developed methods for estimating the force/moment acting at the tip of continuum robots by using measurements of joint-level actuation forces and extrinsic information regarding the type of interaction with the environment.[61,62] These methods have been used in [63] to enable force-controlled exploration of flexible anatomy in a manner that benefits from the palpation methods presented in this research.

Mitra and Niemeyer[48] originally introduced the concept of MMT for the general case of bilateral teleoperation under large communication time delays. Based on this approach, Xia et al.[64] demonstrated model-based telemanipulation for satellite servicing using hybrid force/motion control to accommodate environment mismatch with the slave robot. Force-controlled exploration has also been examined previously as a means of gathering information for registration and updating a pre-operative model.[65] Constrained Kalman filtering was employed to obtain the rigid registration of the model using contact and estimated stiffness information.

Mavash et al.[59] reported a method for stiffness estimation of a tissue model based on discrete palpation. They developed a control system for the da Vinci surgical system that provided force feedback with a position-position controller with friction and

inertia compensation. In more recent work, methods to estimate forces acting at the tip of continuum robots were developed using joint-level actuation forces.[61,62] These methods have been used to enable force-controlled exploration of flexible anatomy.[63]

Surface information is useful in the registration of preoperative data and in providing "augmented reality" information support to the surgeon. Many authors have reported computer vision methods for estimating surface geometry, *e.g.*, [66–69]. Discrete probing can be used to obtain additional information at a large number of points, which can be used to correct for camera-to-robot misalignment. While discrete probing can generate a model of tissue stiffness across an organ, the process can be very time-consuming. In an effort to improve the efficiency of probing for detecting tumors, Ayvali et al.[70] explored Bayesian optimization strategies to guide the probe towards unexplored regions that would result in maximum information gain in predicting stiff regions.

Caccamo et al.[71] developed an online probabilistic framework for autonomous estimation of a deformability distribution map of heterogeneous elastic surfaces from few physical interactions. Caccamo's framework combines both visual and haptic measurements with active exploration and builds deformability maps. There are other vision-based algorithms available for online surface reconstruction.[72] In more recent work, Garg et al.[73] have presented an autonomous tumor localization technique using Gaussian Processes (GP) adaptive sampling. The technique uses a palpation probe to estimate surface stiffness using discrete probing, and an implicit level-set upper

confidence bound (ILS-UCB) algorithm is used for an offline estimation of the tumor boundary.

Limitations : Even though these challenges have been tackled individually, existing approaches have limitations when implemented together in a real-time interactive environment, and require a system infrastructure that is non-trivial to implement. Further, when the system has to perform interactively in response to new incoming data, there are additional problems that arise, such as registration and update of the situational model. To estimate surface information, researchers have used discrete probing strategies,[74] which can be extremely time-consuming and wasteful.

This work presents a component-based framework that addresses the above-mentioned challenges together. In particular, the computational framework facilitates interactive and online updates of the situational model using information from multiple sources while concurrently updating task-based virtual fixtures based on information obtained during manipulation. Further, the CSA framework dynamically corrects for discrepancies between the situational model and the target anatomy.

1.3 Thesis Story

The proposed CSA can be summarized into two fundamental concepts;

(a) *Manipulation is mediated by a situational model.*

(b) *The situational model is updated based on information from manipulation.*

Thesis statement : Complementary Situational Awareness in a telerobotic surgical assistant system helps improve the execution of a complex surgical task, while simultaneously evaluating and monitoring the changes in the task environment using elaborate guidance and sensory capabilities.

Significance : Although robot-assisted surgery addresses many limitations of traditional surgery, it has its own share of disadvantages over traditional surgery. In the most basic form of teleoperation, the slave robot follows a motion command given by the master device operated by a human. Due to uncertainty in the environment, performing complex manipulation tasks using basic teleoperation can be risky and is completely dependent on the surgeon's visual perception of the environment. In some cases, it is impractical or extremely difficult to visualize the anatomical features. The proposed framework aims at dynamically providing information of the situational model to the surgeon, ensuring that the surgeon is aware of the constantly evolving model of the target anatomy.

Basic Requirements : To provide CSA, the computational framework must incorporate many machine capabilities, including sensor information fusion, enforcing operational constraints for human-robot task execution, providing sensory feedback of the situational model environment, and updating an *a priori* model based on the environment. Further, these functions must be executed simultaneously at interactive and near-to-video frame rates.

1.4 Contributions

Key contributions presented in this dissertation are as follows:

- A high-fidelity framework that provides real-time Complementary Situational Awareness while performing complex MIS procedures. [Outlined in **Chapter 2**, discussed throughout]
- Novel offline technique to estimate tissue stiffness and surface geometry using Gaussian Processes. [**Chapter 3**]
- Real-time estimation of tissue stiffness and surface geometry using spatial hash grids and local Gaussian Processes. The estimation is updated graphically in near-video frame rates to visualize for the user/surgeon. [**Chapter 3**]
- Incremental deformable registration technique using spatial hash grids and coherent point drift registration algorithm. [**Chapter 3**]
- Model-mismatch correction using the concept of a proxy slave. When dealing with model-mediated teleoperation model-mismatch is imminent and important to address. [**Chapter 4**]
- Capability of incorporating virtual fixtures and other guidance behaviors for robot assistance. [**Chapter 5, 6**]
- Integration of multiple sensing modalities. [**Chapter 6**]

1.5 Published Work

Material from this dissertation has appeared in the following publications:

1. P Chalasani, A Deguet, P Kazanzides, RH Taylor, "A Computational Framework for Complementary Situational Awareness (CSA) in Surgical Assistant Robots," in 2018 Second IEEE International Conference on Robotic Computing (IRC), 9-16

2. P. Chalasani, L. Wang, R. Yasin, N. Simaan, and R. H. Taylor, "Preliminary evaluation of an online estimation method for organ geometry and tissue stiffness," in 2016 IEEE Robotics and Automation Letters, vol. 3, no. 3, pp. 18161823.

3. L. Wang, Z. Chen, P. Chalasani, R. M. Yasin, P. Kazanzides, R. H. Taylor, and N. Simaan, "Force-controlled exploration for updating virtual fixture geometry in model-mediated telemanipulation," in 2017 Journal of Mechanisms and Robotics, vol. 9, no. 2, p. 021010

4. P. Chalasani, L. Wang, R. Roy, N. Simaan, R. H. Taylor, and M. Kobilarov, "Concurrent nonparametric estimation of organ geometry and tissue stiffness using continuous adaptive palpation," in 2016 IEEE International Conference on Robotics and Automation (ICRA), May 2016, pp. 4164-4171

5. Z. Chen, A. Malpani, P. Chalasani, A. Deguet, S. S. Vedula, P. Kazanzides, and

R. H. Taylor, "Virtual fixture assistance for needle passing and knot tying," in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 23432350.

Chapter 2

Complementary Situational Awareness (CSA)

Unlike in the traditional open surgery, surgeons lack surgical situational awareness during MIS procedures. Although the endoscopic view enhances the visualization of the anatomy, perceptual understanding of the environment and anatomy is still lacking due to the absence of sensory feedback. The CSA framework is developed to address these limitations by providing additional sensory information complementary to the user's awareness of both the remote environment (situational model) and the robot's operational constraints. Based on the interaction with the situational model, CSA not only informs the user about the changes in the task environment but also provides task-specific guidance to assist the surgeon.

This chapter gives a high-level overview of the proposed CSA framework and pro-

vides hardware and software dependencies needed to incorporate CSA to any robotic platform. Detailed descriptions of the behavior and significance of various components of the framework are discussed in following chapters.

2.1 Motivation

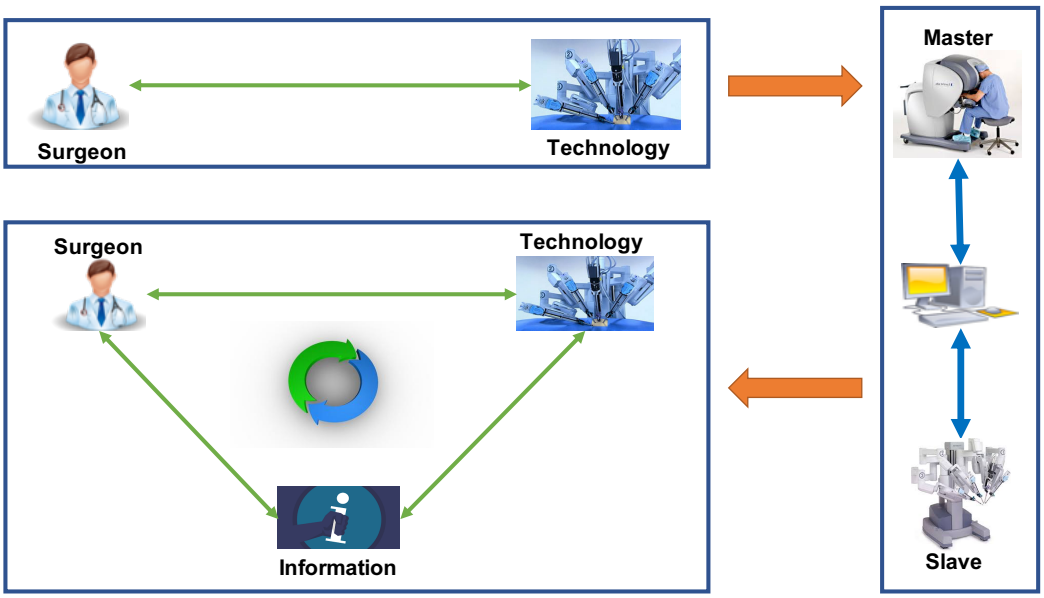


Figure 2.1: Transformation to a three-way communication without CSA

Human-machine partnership requires a two-way communication between the surgeon and the technology. However, if we put a computer between these two devices, it becomes a three-way communication between the surgeon, technology and information. Surgeons use the technology and information to treat the patient better. Figure 2.1 shows the transformation of a two-way communication to a three-way communication with the addition of a computer interface. This computer interface

supplements the perception of the surgeon with more information about the surgical task environment and provides guidance, accordingly. CSA can even further enhance the communication and flow of information as shown in Figure 2.2, by 1) modeling the task environment in real-time, 2) providing haptic feedback based on the updated model, 3) displaying latest surface information on the console view for the surgeon to locate and palpate the abnormalities in the tissue. Further, the framework also minimizes the surgeon's work in various surgical tasks.

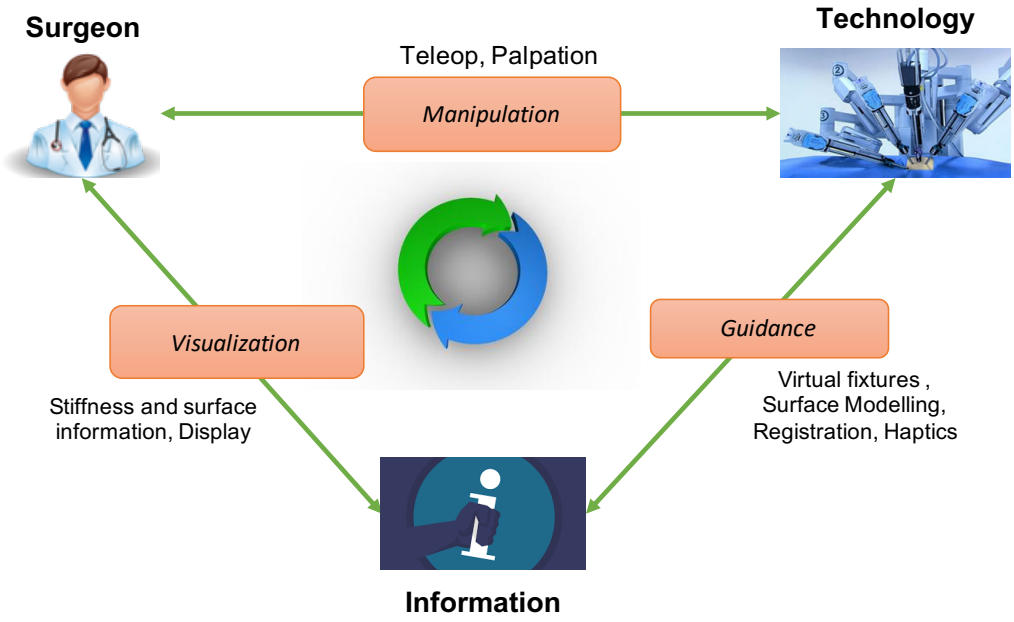


Figure 2.2: Three-way communication with CSA

Figure 2.3(a) depicts a simple scenario of a traditional telerobotic human-machine partnership where sensory information from various components are provided to the surgeon using computer intervention. However, processing of this multi-modal information is still done by the surgeon while performing the surgical task, which con-

siderably increases the workload of the surgeon during the operation. The aim of CSA is to offload the work of the surgeon by providing guidance strategies based on the multi-model sensory information. Figure 2.3(b) shows the modified human-machine partnership with incorporation of CSA. Here, the surgeon manipulates the model of the task environment using task-specific guidance virtual fixtures. Based on the contact information with the model, smooth haptic feedback is provided for the user along with necessary task-specific information displayed on the console view. This helps the surgeon focus on the surgical task and lets the machine figure out the logistics of manipulation and guidance.

The main theme of CSA is as follows:

(a) The surgeon manipulates the model of the task environment,

(b) Manipulation provides information about the task environment, and

(c) The information is then used to update the model of the task environment.

By successfully following these three steps in real-time, the CSA framework can improve the quality of the task execution and decrease the degree of expertise required of a surgeon to perform a complex MIS procedure.

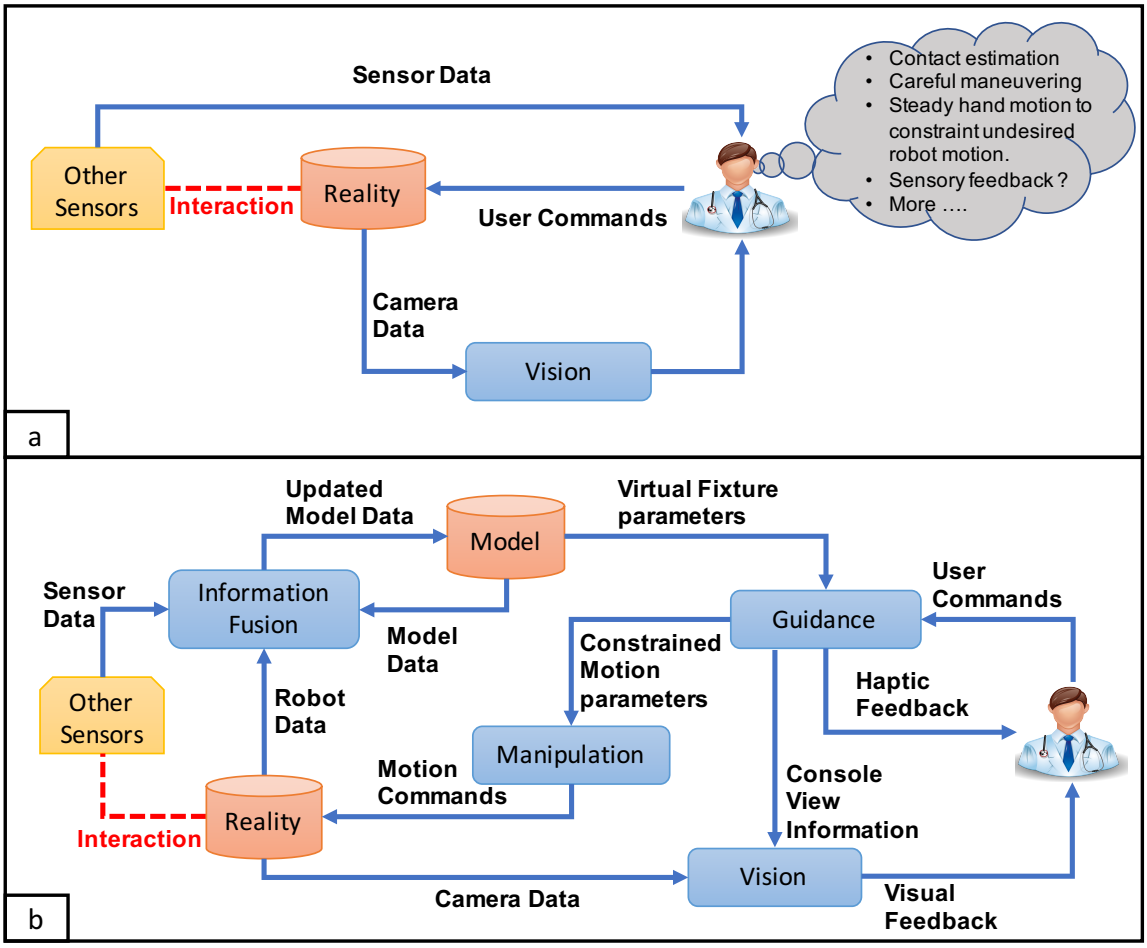


Figure 2.3: a) Conventional human-machine partnership in teleoperation, b) Modified human-machine partnership in teleoperation integrated with CSA

Clinical Motivation

Some of the key surgical tasks that necessitate the development of CSA include

- Palpation: Palpation helps surgeons feel the sensitive anatomy and correlate the actual anatomy with the preoperative data. Examples include identification of arteries during dissection,[75] identification of hepatic aneurysms,[76] intramedullary fixation of tibia and femur during orthopedic surgery,[77] and identification of laryngeal nerves during thyroid surgery.[78]

- Dissection: Dissection around arteries and sensitive anatomy is a critical surgical task for preserving organ function. Examples include dissection around vessels during radical prostatectomy,[79] dissection while preserving the common hepatic duct during cholecystectomy,[80] pelvic lymph node dissection to check for cancer,[81] and selective dissection of renal arteries during partial nephrectomy.[82]

- Ablation: Performing an ablation task can be challenging due to the flexibility of the underlying anatomy. Examples include atrial fibrillation ablation around the pulmonary veins,[83] thermal ablation to remove kidney tumors, inoperable pulmonary nodules and breast tumors,[84] and percutaneous radiofrequency ablation of hepatic tumors, adrenal tumors and adreno-cortical carcinoma metastases.[85,86]

- Separation of adherent organs: This is an important task during abdominal and thoracic procedures. For example, surgeons must separate the gallbladder off

the liver bed during cholecystectomy.[87] Other examples include surgical separation of scar tissue (adhesions) that typically form after abdominal, thoracic,[88] pelvic,[89] and gynecologic surgery.[90]

During robot-assisted and computer-aided surgeries, surgeons attempting to perform a surgical procedure are challenged by the above-mentioned critical tasks. We believe the CSA framework will benefit these applications by providing necessary task-specific constraints to navigate the instrument on a fixed trajectory for ablation, provide information about the tumor location and help localize the tumor boundary, and also provide guidance fixtures for precise resection without damaging the nearby tissue. The framework also provides, and supports, various palpation strategies and sensing modalities to provide information on the intraoperative anatomy to the surgeon.

Example Surgical Workflow

Here we present a sample surgical scenario and discuss how the surgeon can benefit from various functionalities provided by the CSA framework, during a surgical task. For this, we chose to take up a robotic task in renal surgery, in particular, partial nephrectomy.[91] This is a minimally invasive technique to remove a small renal tumor while preserving the remainder of the kidney. The retroperitoneal approach to robotic partial nephrectomy (RPN) is ideal for posterior tumors, which would otherwise require complete mobilization if using a transperitoneal technique.[92] Some tasks

involved in this procedure, that can benefit from the CSA architecture, include hilar dissection, tumor identification, and tumor excision.

During hilar dissection, it is important to stretch the kidney, to improve identification of the vessels and to facilitate dissection of the hilar vessels. Failure to identify the vessels rapidly and correctly can lead to increased duration of operations, the risk of the vessel injury, and the probability of open surgery. A robotic ultrasound transducer can be used to quickly identify the vessels before clamping. The probe's contact force will be monitored by the CSA, thus allowing the surgeon to quickly scan through, without worrying about damaging the tissue. The probe may also be used to confirm sufficient parenchymal ischemia after clamping.

For identifying the tumor, the surgeon can perform a continuous robotic palpation, that can display the estimated stiffness map of the organ. Based on the relative stiffness provided, the surgeon can make an informed decision on identifying the tumor. To further confirm the location of the tumor, the surgeon can switch to an intraoperative ultrasound probe, as before, that will provide segmented stiff regions if any.

Once the tumor is located, the surgeon can make a virtual boundary around the tumor. This will generate a constrained motion for the slave arm to stay on the predefined boundary for excision. Simultaneously, compliance forces will be applied to the master handle preventing the surgeon from deviating from the predefined tumor boundary. The surgeon would be able to redefine the boundary at any time, if needed,

based on the updated stiffness and segmented ultrasound information, which will always be available for viewing.

2.2 System Architecture

CSA is implemented as a component-based framework, using the open source *cisst* libraries, developed at Johns Hopkins University,[93,94] with support for the ROS.[95] Figure 2.4 shows the proposed framework and various component interactions to facilitate CSA. *Teleop* is the central high-level component managing communication between various components of the framework. The *Master Mid Level Controller (MLC)* sends position to the *Teleop*; the *Slave MLC* receives these position commands from the Teleop and executes the motion. Each component in this framework is responsible for a particular task and together they provide situational awareness for the surgeon. This computational framework can be incorporated into any telerobotic platform with the following capabilities: a) The *Master Low Level Controller (LLC)* can be commanded in torque and position control, b) The *Slave LLC* can be commanded in position control and c) there exists a means to report interaction forces.

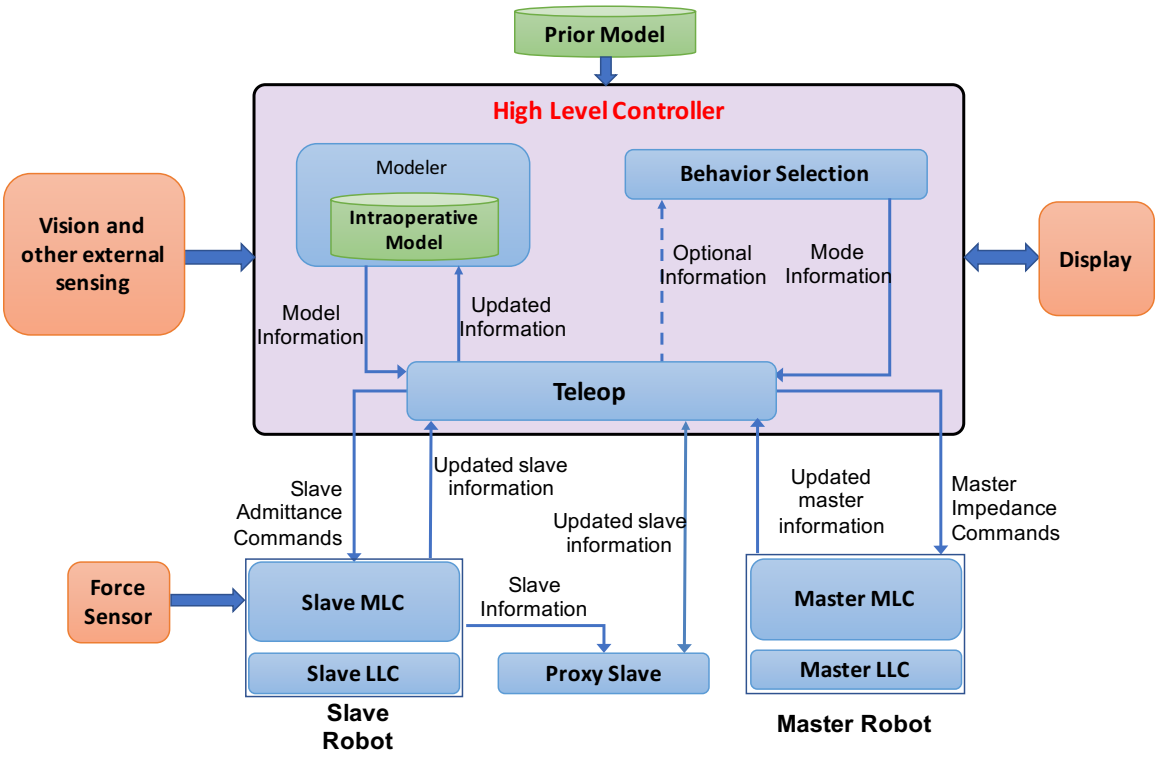


Figure 2.4: Various components in the CSA framework.

Following is a brief description of the various components and their significance to provide CSA in surgical assistant robots :

- The *Teleop* process is the central control point for the system. This process runs as a real-time, clock-driven process (currently, at 200-300 Hz). It is responsible for real-time telemanipulation behavior, and also for managing communications between other components of the system namely: the *Master Controller*, *Slave Controller*, *Proxy Slave Controller*, *Modeler*, and the higher-level *Behavior Selection* processes. The *Teleop* process receives state information from the *Master MLC* and *Slave MLC* and passes this information on to the *Modeler* and the *Behavior Selection* Process. Based on the entire combined state information (master, slave, model, etc.) and the current behavior mode, the *Teleop* component determines appropriate admittance and virtual fixture commands and sends them to the *Slave Controller*. Similarly, appropriate impedance commands are also sent to the *Master Controller*.

- The *Master Controller* process is responsible for the control of the master manipulator. This process consists of two sub-processes: a *Master MLC*, which communicates with the *TeleOp* process, and a *Master LLC*, which communicates with the master manipulator and performs basic joint-level servo control functions. The *Master MLC* runs as a clock-driven process at a repetition rate of approximately 100-500 Hz. The *Master LLC* typically may run at a faster duty cycle. Most of the hardware-dependent components will reside in the

Master LLC, so it should be possible to adapt the system to use other master manipulators in a fairly straightforward manner.

The *Master MLC* receives impedance specification commands from the *Teleop* process and translates them into an appropriate form for execution by the *Master LLC*. The *Master MLC* process also returns state information to the *Teleop* process, including positions and velocities in joint and cartesian spaces, gripper state of the master manipulator, and forces and torques exerted by the master robot on the surgeon's hand.

- The *Slave Controller* process is responsible for the control of the slave manipulator. Like the *Master Controller*, this process consists of a *Slave MLC*, which communicates with the *Teleop* process, and a *Slave LLC*, which communicates with the slave manipulator. The *Slave MLC* runs as a clock-driven process at a repetition rate of approximately 100-500 Hz. The *Slave LLC* typically may run at a faster duty cycle (such as 1kHz). The *Slave Controller* also contains a *Force Sensing* component, which may be implemented as part of the *Slave LLC* or as a separate process, depending on the hardware configuration. Almost all of the hardware dependencies will be managed in the LLC, although there may need to be some changes in the MLC as well to accommodate special hardware needs.

  The *Slave MLC* receives admittance and virtual fixture specification commands

from the *Teleop* process and translates them into Cartesian or joint position/velocity commands that are passed onto the *Slave LLC*. In some embodiments, the MLC may pass on specialized force admittance commands to the LLC, although this function would normally be performed in the MLC. The *Slave MLC* run loop transforms admittance and virtual fixture specifications into a constrained quadratic optimization problem, which also may comprise manipulator-specific constraints such as joint position, velocity, and acceleration limits.

The *Slave MLC* receives state information from the LLC, combines this information with other *Slave Controller* information (e.g., forces and contact information) and passes the combined state information back to the *Teleop* process.

- The *Proxy Slave* implements pure position control representing *Teleop*'s notion of where the slave robot should be. The situational model in the virtual environment is registered to a simulated slave robot, called the *Proxy Slave*. The user teleoperates the proxy slave robot and based on the interaction with the situational model *Teleop* sends appropriate commands to the *Master* and *Slave MLC*. The advantage of the proxy slave robot is that it provides the state information of the slave robot with an ideal position control. This allows the *Teleop* component to use the contact information of the proxy slave robot with the situational model for haptic rendering on the master side. Additionally, TeleOp uses this information to send admittance commands to the slave side for force limiting purpose and additional task-based constraints.

- The *Behavior Selection* process runs in the background and communicates with the *Teleop* process to inform it of changes in the desired behavior (e.g., simple unilateral telemanipulation with no force-feedback, bilateral model-mediated telemanipulation with or without force limitation, bilateral telemanipulation with superimposed palpation motion). It receives state information from the TeleOp process and the Modeler, as well as direct input from the user.

- The *Modeler* process is responsible for maintaining a model of the task at the environment side. Initially, based on some pre-operative information this model will consist of a surface mesh representation of an anatomic organ or phantom object, annotated with stiffness information for each triangle, together with registration information giving the pose (position and orientation) of the model relative to the slave robot. The *Modeler* receives state information from the *Teleop* process and combines this information with a prior model (such as what might come from a CT scan) to produce an updated model. The updated model is then passed on to the other components in the system, along with the estimated confidence level of the Modeler with respect to various components of the model.

2.3 Development Dependencies

The CSA framework is a layer that can be employed on top of any telerobotic platform. For development purposes, the dVRK platform was chosen as the principal experimental telerobotic platform due to the familiarity to the architecture. In Section 6.3, the integration of the CSA with the UR3 robot, as a secondary platform, is also discussed, along with the details of experimental implementation. Following are few of the principal software library dependencies of the dVRK and CSA platform:

cisst-saw

The dVRK uses C++ on Linux, though most of the software is portable to other platforms. It is a real-time, component-based framework that enables low latency control. The dVRK is based on the open source *cisst* libraries developed at Johns Hopkins University (JHU).[93,94] *cisst* is a collection of various open-source libraries which enables one to develop off-the-shelf application components called Surgical Assistant Workstation (SAW) packages. The dVRK uses several of these SAW components, along with open-source electronics to create a research platform.

CSA is developed in a similar way using several SAW components. Some of the key *cisst* libraries that are used in the dVRK as well as in the CSA platform are detailed below for better understandability:

- *cisstVector* : This is a collection of classes used for linear algebra and spatial

transformations. This includes vectors, matrices, multi-dimensional arrays and some classes to represent 2D and 3D transformations. *cisstVector* is heavily templated for flexibility and efficiency.

- *cisstParameterTypes* : These are a collection of strongly-typed payload datatypes to facilitate data exchange between *cisst* components.

- *cisstMultiTask* : This library defines the *cisst* component model that is implemented by the *mtsComponent* base class. It provides all the mechanisms to create interfaces (explained below) but does not handle any threading. To ensure mechanisms of thread safety when communicating through interfaces, the component needs to derive from the *mtsTask* base class. Several types of components are derived from this class. However, only two of key classes that are used in CSA are detailed here:

  1. *mtsTaskPeriodic* : This is a base class for any task that needs to be executed periodically.

  2. *mtsTaskContinuous* : This is a base class for any task that needs to be executed as fast as possible.

  All tasks derived from *mtsTask* define three pure virtual methods, *Startup(void)*, *Cleanup(void)* and *Run(void)* methods that help initialize, cleanup and execute, respectively. The *Run* method is where the core logic is implemented for continuous/periodic execution.

cisstMultiTask follows a "command pattern" paradigm, i.e., an object is used to encapsulate all the information needed to perform an action or trigger an event at a later time. In this design, an object Application Programming Interface (API) is defined by a list of pointers on methods. This list of method pointers (commands) is queried using string comparison during run-time rather than compile-time binding. Commands are grouped in "provided interfaces" corresponding to the list of provided functionalities acting as a server. Similarly, a list of required functionalities by a client component are grouped in a "required interface" on the client side. For the client component to use the functionalities provided by the server component, its required interface needs to be connected to the interface provided by the server. Figure 2.5 illustrates a sample interface connection where the required interface of component B is connected to the provided interface of component A, and the required interface of component C is connected to the provided interface of component B.

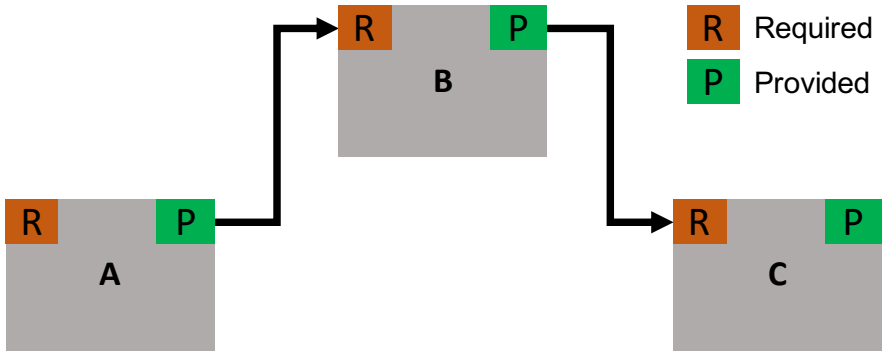


Figure 2.5: Sample interface connection

All the messages received by threaded components are queued for thread safety. The receiving component typically calls *ProcessQueuedCommands* and *ProcessQueuedE-*

vents at the start of the *Run* method to dequeue all commands of all provided interfaces and all events of all required interfaces, respectively.

ROS Bridge

Most of the existing robotic research platforms use ROS to develop high-level interfaces. Thus, having the CSA framework to be ROS-compatible was necessary to ensure that the developers can utilize CSA functionality over a ROS channel. This is possible due the availability of a *cisst-ROS* bridge used in the dVRK that the CSA framework uses to provide access to CSA functionality using MATLAB-ROS or Python-ROS. Further, *rviz*, which is a 3D visualizer for the ROS framework, is used for displaying model information and task-specific data. This display window is placed towards the bottom-left of the master console view without obstructing the main view.

Other Library Dependecies

Other library dependencies that are used in the dVRK and the CSA platform include:

- **Qt :** Qt is a cross-platform application and user interface framework. Developers have the flexibility to create their application level interfaces in C++ or python.

- **PCL :** Point cloud library (PCL) is a standalone open project for 2D/3D image and point cloud processing.

2.4 Contributions

The principal contribution reported in this chapter is:

1. System and architecture designs of CSA such that it can be readily incorporated into any existing robotic platform that can meet certain hardware requirements.

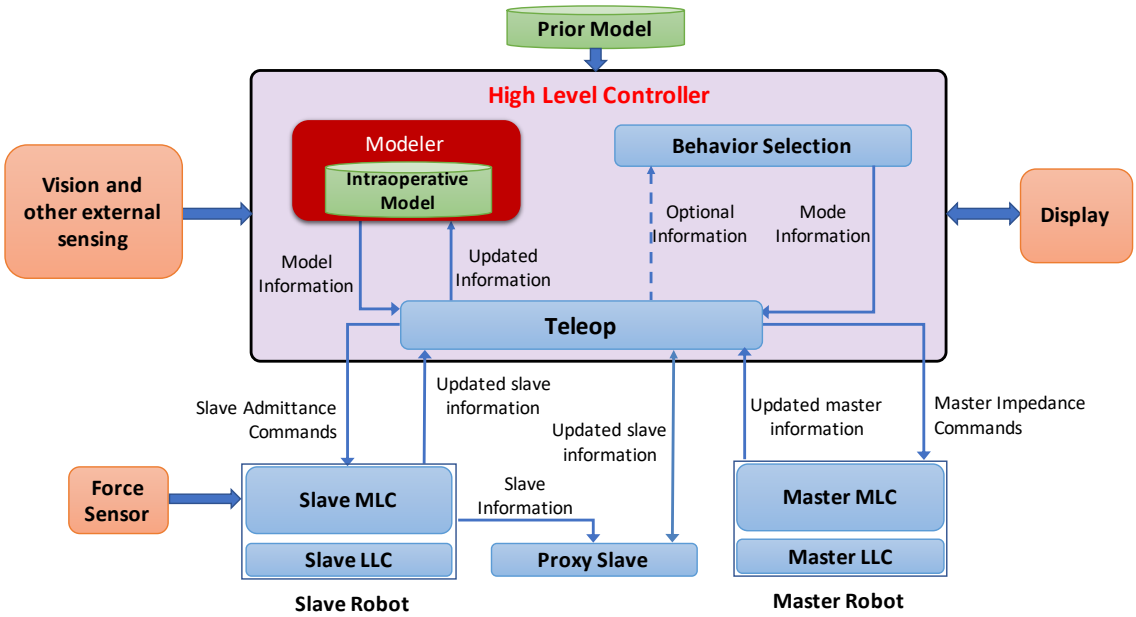
2.5 Published Work

Material from this chapter has appeared in the following publication:

1. P Chalasani, A Deguet, P Kazanzides, RH Taylor, "A Computational Framework for Complementary Situational Awareness (CSA) in Surgical Assistant Robots," in 2018 Second IEEE International Conference on Robotic Computing (IRC), 9-16

Chapter 3

High-Level Controllers - *Modeler*



In MMT, the user controls the master device to manipulate the model of the task environment and based on such interactions with the model, appropriate commands are sent to the slave robot to manipulate the anatomy. However, manipulation is

only possible if the model of the task environment is thoroughly registered to the master device based on the registration transformation between the slave robot and the anatomy. Another key feature of MMT is the provision of haptic feedback, which requires the model of the task environment to be dynamically updated in real-time so that the user experience accurate haptic feedback. This is essential during a surgical task since surgeons generally deal with non-rigid and deformable objects.

Lets consider a simple case of palpation of the liver during traditional open surgery; in a traditional surgical operation, the surgeon has to first make a large incision to gain access to the liver and, then the surgeon gently apply pressure on the surface of the liver using his or her hand/fingers to determine the condition of an underlying tissue. In a MIS procedure, however, the surgeon would not have direct access to the tissue and would need some kind of haptic/visuo-haptic feedback to improve their telepresence. One possible approach to provide haptic/visuo-haptic feedback is through the characterization of tissue properties, based on which a map of tissue properties can be provided to the surgeon. To characterize tissue properties, the control architecture needs to continuously monitor the deformation of the model based on the contact information of the tool and the anatomy. This is done by adding a force sensing modality in the task environment to analyze the tool-tissue interactions, and then reconstructing organ geometry and tissue stiffness based on preoperative data and current contact information. Further, the estimated geometric information and tissue stiffness are used to perform a deformable registration technique to constantly

update the model of the anatomy for a functional MMT when dealing with non-rigid objects.

Modeler is one of the key components of CSA that deals with the aforementioned problem of maintaining a sensible model of the task environment for a functional MMT and accurate haptic feedback. *Modeler* is responsible for providing the *Teleop* component with the latest model information based on the interaction between the slave robot and the anatomy. Additionally, the component also provides a means of communicating with an optimization framework that provides the next optimal location for exploration to maximize information gain.

3.1 Prior Art

There have been studies to estimate stiffness properties of an organ based on tool-tissue interaction force information. Mavash et al.[59] reported results of stiffness estimation of tissue based on discrete palpation. They developed a control architecture for the da Vinci surgical system that provides force feedback through a position-position control schema along with friction and inertia compensation. In more recent work, methods to estimate forces acting at the tip of continuum robots were developed using joint-level actuation forces.[61,62] These methods have been used to enable force-controlled exploration of flexible anatomy.[63] Garg et al.[73] have presented an autonomous tumor localization technique using Gaussian Processes adaptive sampling.

The technique uses a palpation probe to estimate surface stiffness using discrete probing and an implicit level-set upper confidence bound (ILS-UCB) algorithm is used for an offline estimation of the tumor boundary.

All the above-mentioned studies on stiffness and organ geometry estimation depend on discrete probing strategies. One advantage of discrete probing is that it permits accurate assessment of tissue stiffness and surface location at the point probed. The drawback is that it can be inefficient, compared to continuous palpation primitives. The need for continuous palpation primitives required a novel implementation for estimating organ geometry and tissue stiffness. The control law and specifications of the palpation primitives are discussed in Chapter 5.

As mentioned earlier, the sole purpose of the *Modeler* is to make sure the model of the task environment is always updated based on the contact information of the slave robot tool with the task environment. This task is divided into various subprocesses (Figure 3.1):

- Extract surface information **(Section 3.3)**
- Using the surface information to update the task model. **(Section 3.4)**
- Provide optimal trajectory or direction to maximize the information gain based on the updated model and estimated surface stiffness information. **(Section 3.6)**

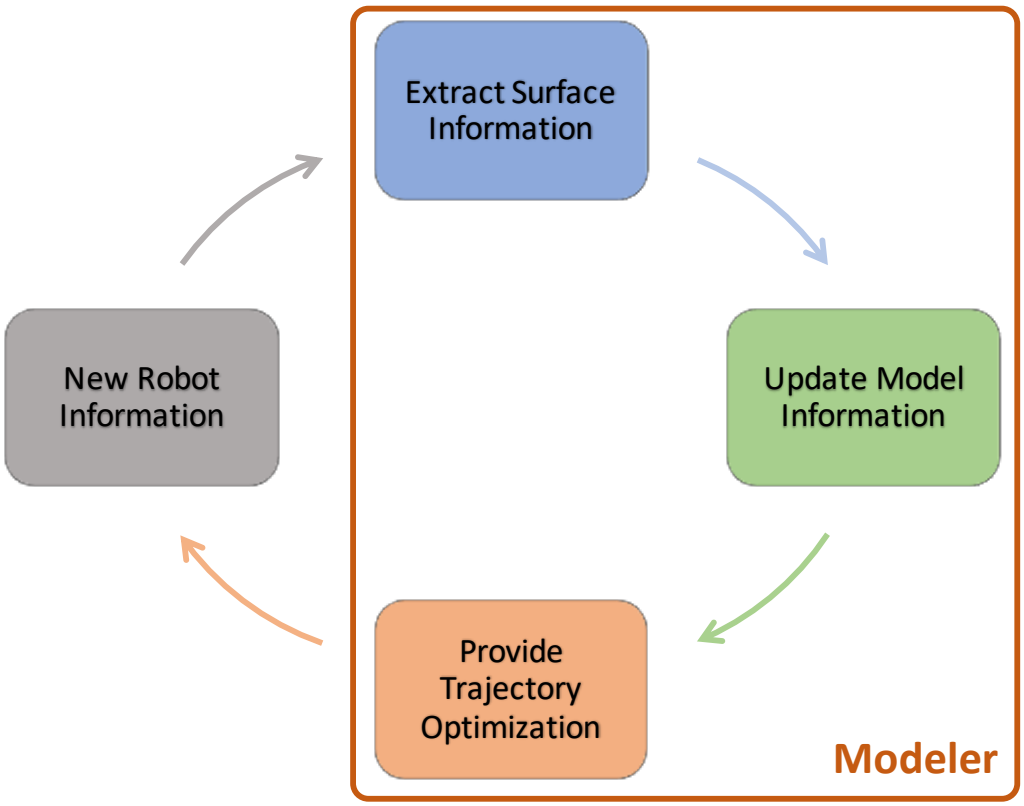


Figure 3.1: *Modeler* subprocesses

3.2 *Modeler* Workflow

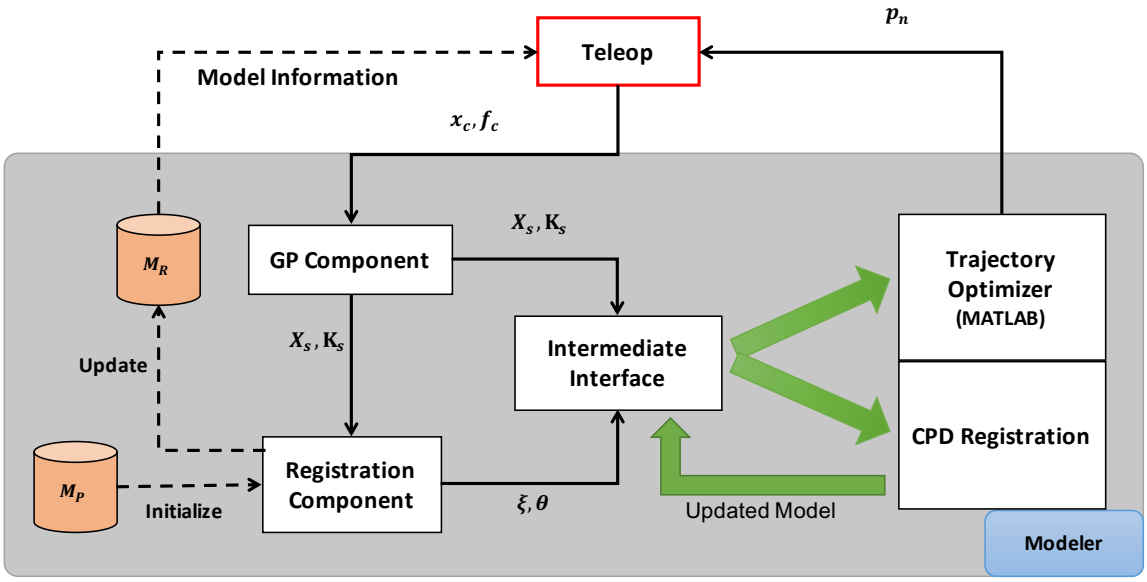


Figure 3.2: *Modeler* Workflow

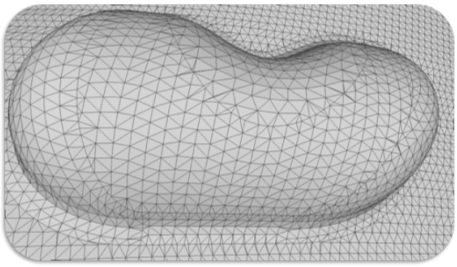
Figure 3.2 shows the communication flow between the *Modeler*'s subprocesses. Based on the current model data, the *Teleop* (described in Chapter 4) component provides the contact information, i.e, the current contact location of the tip of the slave robot with the tissue based on the robot kinematics (x_c), and the contact force sensed by a force sensor (f_c). This information is passed to the first module called the "GP Component", which uses the current and stored contact information to estimate the surface point (x_s) and its corresponding tissue stiffness (κ_s) at the current surface location. The current surface information, along with the previous geometry and stiffness estimates, is sent to the second module called the "Registration Component" for the purpose of a model update. The *Registration* component uses the estimated sur-

face geometry based on contact information and preoperative model (M_P) to perform a rigid registration step using the Iterative Most Likely Point (IMLP) algorithm.[96] The outputs of this module are the registration parameters (θ) and an intermediate model data (M_I), which is registered to align as closely as possible with the preoperative data. However, during surgery, the intraoperative model deviates from the preoperative model due to tissue deformation. Thus, after the initial alignment, the model information is passed over to MATLAB via an "Intermediate Interface" to perform a non-rigid registration step based on the Coherent Point Drift (CPD) algorithm[51]. CPD performs well if the angle of initial misalignment is less than 70 degrees, thus we perform an optional rigid alignment before CPD. The result of CPD is an updated model (M_R) that is fed back to the *Teleop* component. The user then manipulates on the updated model of the task environment and this runs in a loop.

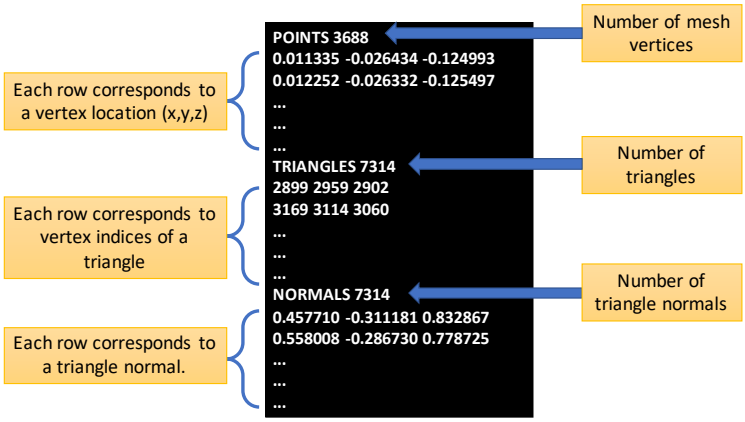
The architecture is developed in a modular manner such that, for a tissue exploration task, developers are able to attach their "Trajectory Optimizer" setup to use the estimated stiffness and other model information to provide an optimal trajectory or direction to maximize information gain. All the model information is sent over to MATLAB and developers have access to necessary ROS functionalities to send information from the optimized trajectory back to the robot.

Data Format of the Model

A mesh data format was used to represent a model of the task environment. A custom file format was used to load the mesh data into the framework. Figure 3.3(a) shows an example model of a kidney phantom used in the experiments, described in Chapter 6. Here the kidney model is represented by a set of vertices and triangles, where this information is stored in a file as shown in Figure 3.3(b).



(a) Triangular mesh of a kidney model



(b) Mesh file format

Figure 3.3: Data representation and storage formats of a phantom model in CSA

Model Assumptions

Throughout this dissertation, three simplifying assumptions are made regarding the model, which are reasonable for semi-rigid organs such as kidney or liver :

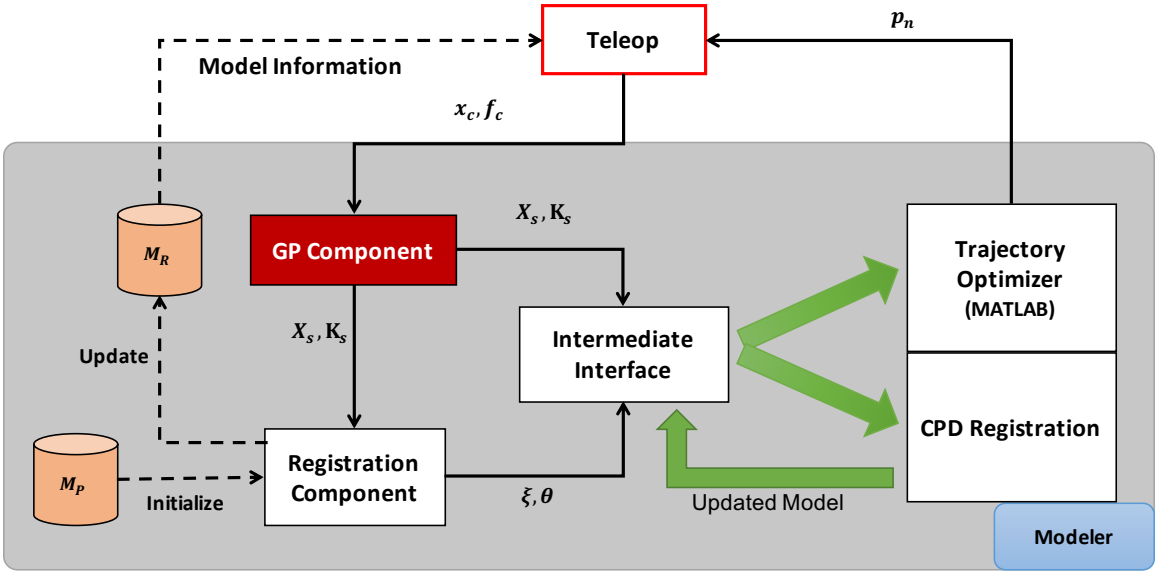
1. The direction of the force vector at a point under the tissue surface is essentially equal to the surface normal at that location. Thus, surface stiffness can be computed using the normal component of the tool-tissue interaction force.

2. Stiffness can be computed using a simple linear force model, i.e., $\vec{F} = \kappa\vec{x}$ where $\vec{F}$ is the force vector, κ is stiffness and $\vec{x}$ is displacement.

3. The target organ is rigid enough so that forces applied at one point deform the organ only a small region around the probe tip.

In the following sections, various components of the *Modeler* will be discussed in further details. First, we discuss the details of estimating surface properties, organ geometry, and stiffness. Analytical results are also presented for comparison between offline and online approach. Using the estimated geometry an online registration framework is discussed with a sample workflow example. Our initial implementation of trajectory optimizer is also demonstrated and results are discussed accordingly.

3.3 Surface Estimation (GP Component)



This process is responsible for providing surface and stiffness information based on continuous palpation. The method is based on GP estimation. The initial implementation explores the application of GP in estimating organ geometry and tissue stiffness in an offline manner. In this approach, we use the GP formulation to generate a predictive distribution of force-field underneath the organ surface. This allows us to predict a force value at any given location underneath the surface. Using this model, we determine the surface height and tissue stiffness. However, for the CSA framework to execute in real-time, the estimation has to be done incrementally. Our new approach is based on learning the local force model around the palpation region using GP in real-time. This model is then used to incrementally estimate the local stiffness and shape of the anatomy while the organ is being palpated. Spatial data

structures, specifically hash grids, are used to store position and force information for efficient data storage and retrieval. This allows for low-latency estimation of local surface information around the region of palpation.

In the next section, some necessary background on GP is provided for the reader to better understand the offline estimation techniques used in the literature, described in the later subsections. Later, we discuss the online approach for near-video frame rate updates of the stiffness map.

3.3.1 Gaussian Processes

A GP is a collection of random variables, any finite number of which have (consistent) joint Gaussian distributions.[97,98] GP is a non-parametric approach where it finds a distribution over all the possible functions and is commonly represented as:

$$f \sim \mathcal{GP}(\mu(x), k(x, x')), \tag{3.1}$$

where it is completely described by its mean $\mu(x)$ and a covariance function $k(x, x')$. This is a natural generalization of the Gaussian distribution whose mean and covariance is a vector and matrix, respectively.

Given n observations, $\bar{y} = \{y_1, \ldots, y_n\}$, each observation from dataset $\bar{y}$ can be assumed to be sampled from an $n - variate$ Gaussian distribution. The prior mean of a GP is often assumed to be zero everywhere. In such cases, what relates one

observation to another is just a *covariance function*, $k(x, x')$. The usual choice of covariance function is the 'squared exponential' kernel, because of being flexible and physically intuitive.:

$$k(x, x') = \sigma_f^2 \exp\left[\frac{-\|x - x'\|^2}{2l^2}\right], \tag{3.2}$$

where σ_f^2 is the maximum allowable covariance and l is the length scale. Since, data is often noisy, due to measurement errors, sensor resolution, etc., Gaussian white noise with variance σ_n is added to the covariance through the Kronecker delta function $\delta(x, x')$:

$$k(x, x') = \sigma_f^2 \exp\left[\frac{-\|x - x'\|^2}{2l^2}\right] + \sigma_n^2 \delta(x, x'), \tag{3.3}$$

To prepare for GP regression, the covariance function stated in Equation 3.3, is calculated using the following three matrices,

$$K = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{bmatrix} \tag{3.4}$$

$$k_* = [k(x_*, x_1) \dots k(x_*, x_n)] \qquad k_{**} = k(x_*, x_*) \tag{3.5}$$

The diagonal elements of K correspond to $\sigma_f^2 + \sigma_n^2$, and the extreme diagonal elements tend to zero when x spans a large-enough domain.

Since the key assumption is that the observations are sampled from a multivariate

Gaussian distribution, the data can be represented as

$$\begin{bmatrix} y \\ y_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K & K_*^T \\ K_* & K_{**} \end{bmatrix}\right) \tag{3.6}$$

Thus, the conditional probability $p(y_*|y)$ can be written as,

$$y_*|\bar{y} \sim \mathcal{N}(k_* K^{-1}\bar{y}, k_{**} - k_* K^{-1} k_*^T), \tag{3.7}$$

or, equivalently, the mean and variance (uncertainty in the estimate) of y_* is

$$\mu_f(x_*|\bar{y}) = k_* K^{-1}\bar{y}, \tag{3.8}$$

$$V_f(x_*|\bar{y}) = k_{**} - k_* K^{-1} k_*^T \tag{3.9}$$

3.3.2 Direct GP modeling

In this work, two modes of estimation were explored using GP. The first imposes a model directly on the force field measurements (described in this subsection) and the second is based on Latent GP estimation (described in Section 3.3.3).

Formulation

Using the formulation defined in Section 3.3.1, the GP model over force field $\tau(x) \sim \mathcal{GP}(\mu_\tau(x), k_\tau(x, x'))$ is defined where each point x corresponds to a palpation

point $p = (p_x, p_y, p_z) \in \mathbb{R}^3$. This model can be used to estimate forces at different depths in the direction of the force vector. Stiffness κ is estimated along the force direction by assuming a linear stiffness model of the form

$$f_2 - f_1 = \kappa(d_2 - d_1), \tag{3.10}$$

where f_2, f_1 are forces at depths d_2, d_1 respectively. The estimated mean (μ_κ) and variance (V_κ) of the surface stiffness in the force direction from an interior point p_{far} given by

$$\mu_\kappa(p_{far}) = \frac{\mu_\tau(p_{far}) - \mu_\tau(p_{close})}{d} \tag{3.11}$$

$$V_\kappa(p_{far}) = \sqrt{\frac{V_\tau^2(p_{far}) + V_\tau^2(p_{close})}{d^2}}, \tag{3.12}$$

where p_{far} is chosen such that it is deep inside the surface. This is because the signal-to-noise ratio of the force sensor gets better as we push into the surface, resulting in an accurate direction of the force vector. p_{close} is a point along the force direction and $d = ||p_{far} - p_{close}||$ is the distance between the two points, p_{far} and p_{close}. Figure 3.4(a) shows an overall illustration of these points.

Next, the surface geometry can be estimated in three different ways :

1. **Surface estimation from stiffness**: In this method, surface geometry is calculated using the estimated stiffness distribution. The mean and variance of

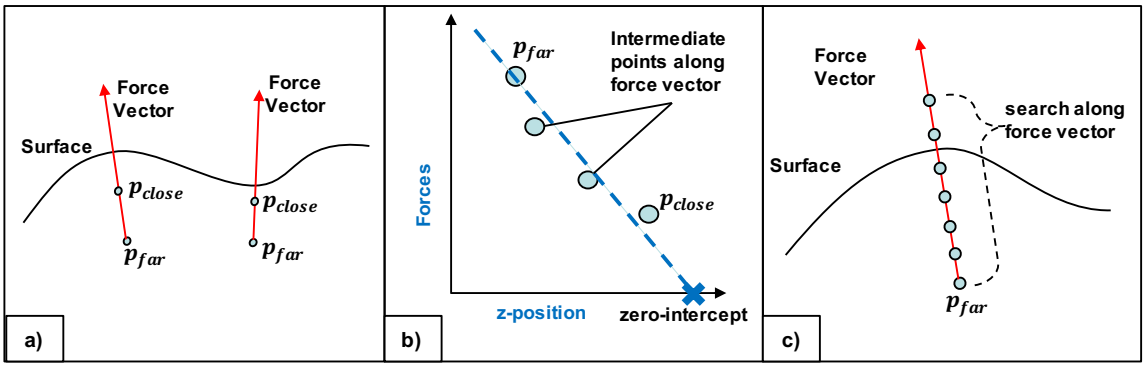


Figure 3.4: Estimation of surface point from the predictive distribution of force field from GP model, a) Illustration of p_{far} (deep point inside the organ) and p_{close} (inside point relatively closer to the surface), b) Zero-crossing, c) Linear search

the surface geometry is calculated as follows,

$$\mu_s(p_{far}) \quad = \quad p_{far} + \frac{\mu_\tau(p_{far})}{\kappa}.$$

Due to the nonlinear relationship between stiffness and surface geometry, the resulting covariance can be approximated[99] according to

$$V_s(p_{far}) = J\Sigma J^T,$$

where

$$J \quad = \quad \left[\frac{1}{\kappa} \quad -\frac{\mu_\tau(p_{far})}{\kappa^2}\right], \Sigma = \begin{bmatrix} V_\tau(p_{far}) & 0 \\ 0 & V_\kappa(p_{far}) \end{bmatrix}.$$

2. **Zero-crossing**:[100] A line is fit through the predicted force and the z-components

of position along the direction of force vector from p_{far}. The intercept value at which the line crosses the axis defining the z-components of the intercept value for which force is zero, is called the zero-intercept. This zero-intercept value is considered to be the surface point (Figure 3.4(b)).

3. **Linear search**: Using the GP force model, a linear search strategy is used to find near-zero mean forces with minimum variances along the force direction of point p_{far} (Figure 3.4(c)).

In the first method, the estimated geometry depends on predicted stiffness, i.e., there is significant error propagation from force to stiffness and from stiffness to geometry. The second and third methods estimate stiffness and geometry directly from the force and position distribution, thus minimizing the error propagation. Therefore, the second method is used in the current implementation, while the third method will also produce a similar result.

Algorithm 3.1 Direct GP

Input: Palpated positions and the corresponding force values, I = $\{\vec{p_i},\ \tau_i\}$

Output: $\mathcal{N}(\mu_\kappa, V_\kappa), \mathcal{N}(\mu_s, V_s)$

$I_s(p_s, \tau) = SubSample(I)$

$X = p_s\ ,\ Y = \tau$

$GPTrain(X, Y)$

$\mu_\tau(p_c), V_\tau(p_c) = GPEstimate(p_c)$

$\mu_\tau(p_{cd}), V_\tau(p_{cd}) = GPEstimate(p_{cd})$

$\mathcal{N}(\mu_\kappa, V_\kappa) = \mathcal{N}(\frac{\mu_\tau(p_{cd}) - \mu_\tau(p_c)}{d},\ \sqrt{\frac{V_\tau^2(p_c) + V_\tau^2(p_{cd})}{d^2}}$

$\mathcal{N}(\mu_s, V_s) = p_{cd} + \frac{\mu_\tau(p_{cd})}{\kappa},\ J\Sigma J^T$

Experiment

The direct GP approach was applied to a palpation set consisting of 77000 palpation points and force pairs. For verification purposes, initially, the true stiffness and geometry were estimated using all the palpation points. However, a subset of n = 2500 points was used for training and a different set of m = 1000 points was used for prediction. For display purposes, the estimated stiffness and geometry were interpolated over a regular grid with 1mm cell size. For this experiment, a length scale of 5mm was used along with an optimized variance parameter, σ_f, by maximizing the negative log-likelihood, as described in [97]. The resultant value of σ_f was 0.86, which complies with the observations because the maximum force applied on the object was around 1.6N and it lies in the 95% confidence level of the maximum variance. The noise parameter was empirically modeled, $\sigma_n = 0.01N$, by analyzing the noise observed in force values measured when various static loads were applied on the sensor. Fig 3.5 shows the ground truth estimation using 77000 samples and Fig 3.6 compares the estimation result with the ground truth. The Root Mean Square (RMS) stiffness error for this particular set of samples was 120.4 mN/mm and the RMS error of the surface geometry was 2.61mm. This accuracy in stiffness/surface estimation is sufficient for a surgeon to detect stiff features or abnormal growth of tissue.

This approach was also tested against varying sizes of input samples. The variance in the force field decreases as the number of training samples were increased. This

makes sense because it is expected to see low uncertainty in the prediction of an unknown observation when the prediction is based on a large number of neighboring known observations. However, RMS error in the surface geometry decreases very slowly as the number of training samples is increased. The reason for this behavior is because of the way we are calculating the surface geometry. The error in surface estimation is not only propagated from the force uncertainty, but also from stiffness uncertainty. An alternative way of calculating the surface geometry, using the force field, is to do a linear search in the vertical direction, at each grid point, and look for the distribution where the mean is zero and has the minimum variance. This will give an estimate of the surface, with a much higher confidence than the one computed using the predictive distribution of stiffness.

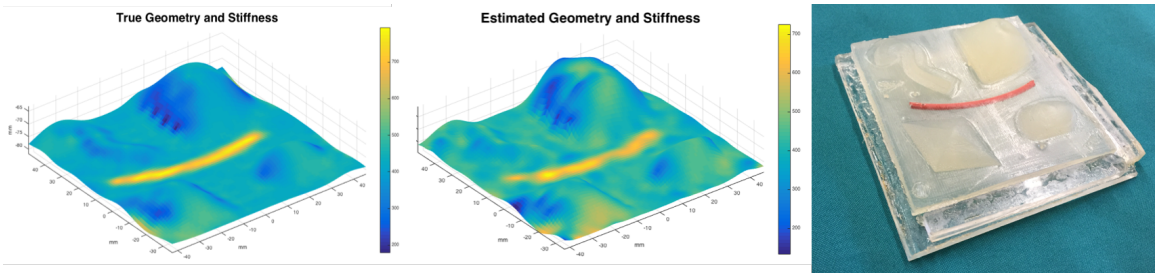


Figure 3.5: True surface stiffness and geometry

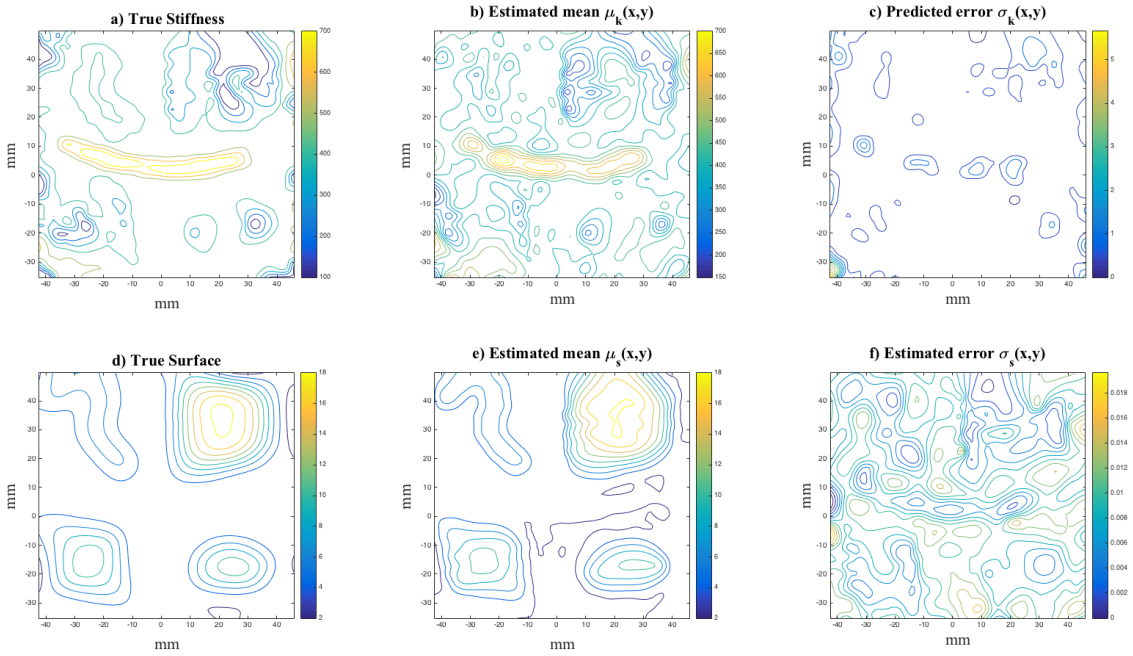


Figure 3.6: Direct GP estimation a) Contour plot of true stiffness; b) Contour plot of estimated mean of stiffness; c) Contour plot of estimated error in stiffness; d) Contour plot of true surface height; e) Contour plot of estimated mean of the surface geometry; f) Contour plot of estimated error in surface geometry

3.3.3 Latent GP modeling

As an alternative, instead of imposing a model directly on the force field measurements, the construction of a model of the underlying physical properties of the palpated object is considered. These properties include the physical geometry and the surface stiffness. A key advantage of such an alternative approach is that prior information about these properties and about their relationship to force measurements could be incorporated as part of the estimation process. This is in contrast to first estimating an unconstrained force field and then deriving the surface and stiffness from the force model.

Our approach is therefore to place a GP model over the surface $s(x) \sim GP(\mu_s(x), k_s(x, x'))$ and over the stiffness $\kappa(x) \sim GP(\mu_\kappa(x), k_\kappa(x, x'))$. Existing knowledge about the stiffness spatial correlation (e.g., the size of the abnormal tissue) and about the variability and roughness of the surface can be captured through proper choice of GP covariance parameters. For simplicity, object geometry is represented as a graph of the height p_z over the 2-D (p_x, p_y) domain. Similarly, the concern here is also with estimating the stiffness near the object surface, which motivates representing stiffness using a graph over the (p_x, p_y) domain.

The key challenge is that there is no direct measurement of stiffness $\kappa(x)$ or surface $s(x)$ at locations $x = (p_x, p_y)$. Therefore, in this approach, the output data of these GPs is treated as *hidden variables* that will be optimally estimated using the *induced measurements* of the forces obtained at 3-D locations p.

Formulation

Here, a general approach to the latent GP estimation is introduced and then, its application to the palpation scenario is demonstrated.

Consider the problem of estimating a vector-valued function $f : \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_y}$, defined according to

$$y = f(x) \equiv \begin{bmatrix} f_1(x) \\ \vdots \\ f_n(x) \end{bmatrix}$$

with inputs $x \in \mathbb{R}^{n_x}$ and outputs $y \in \mathbb{R}^{n_y}$.

Assume that y cannot be directly observed and instead, noisy observations $z \in \mathbb{R}^{n_z}$ can be accessed such that

$$z(x) = h(x, f(x)) + \epsilon(x),$$

where $h(x, y)$ is a known nonlinear function with additive Gaussian noise $\epsilon(x) \sim \mathcal{N}(0, \Sigma_\epsilon(x))$. The goal is to optimally estimate f using a set of M_z collected observation pairs $D_z = \{(x_j, z_j)\}_{j=1}^{M_z}$.

Although $f(x)$ is unknown, some knowledge about the spatial correlation between its inputs and outputs is assumed to be known. This motivates the representation as a Gaussian Process, i.e., $f(x) \sim GP(\mu(x), k(x, x'))$. Note that it can be assumed that all outputs f_i and f_j for all $i \neq j$ are uncorrelated (in which case $k(x, x') \in \mathbb{R}^{n_y \times n_y}$

is diagonal), or that they are coupled[101] (in which case $k(x, x')$ will have off-diagonal elements). Normally, the posterior of such a GP would be constructed given data pairs $D_y = \{x_j, y_j\}_{j=1}^{M_y}$, which in our case are unavailable, or latent. Our proposed approach is to select the points x_j in D_y using some rule (e.g., they could be a subset of those in D_z or otherwise could be placed on a uniform grid over the domain of interest) and to regard the outputs y_j in D_y as unknown variables to be optimally estimated. In addition, the GP hyperparameters θ_{hyper} (e.g., the maximum covariance and length-scale used in standard kernels) could also be treated as unknowns. Thus, a vector of latent variables $\theta = (\bar{y}, \theta_{\text{hyper}})$, where $\bar{y} = (y_1, \ldots, y_{M_y})$, can be formed, and we assume that a prior is available as

$$\theta \sim \mathcal{N}(\cdot|\theta_0, P_0),$$

with a known mean θ_0 and covariance matrix P_0.

The most likely GP model of f can now be estimated using a maximum-likelihood approach by optimizing

$$\begin{aligned} &\min_{\theta} \left[-\log p(f|D_z, D_y) \right] \\ &= \min_{\theta} \left[-\log p(\bar{z}|f, \bar{x}, \bar{y}) - \log p(f|\bar{x}, \bar{y}) \right] \\ &= \min_{\theta} \left[\sum_{j=1}^{M_z} \|h(x_j, f(x_j;\theta)) - z_j\|^2_{\Sigma_\epsilon^{-1}(x_j)} + \|\theta - \theta_0\|^2_{P_0^{-1}} \right], \end{aligned} \tag{3.13}$$

where the GP form of f is given by

$$f(x_j;\theta) = \bar{k}(x_j)^T K^{-1}\bar{y},$$

where

$$\bar{k}(x) = \begin{bmatrix} k(x,x_1) \\ \vdots \\ k(x,x_{M_y}) \end{bmatrix}, K = \begin{bmatrix} k(x_1,x_1) & \cdots & k(x_1,x_{M_y}) \\ \vdots & \ddots & \vdots \\ k(x_N,x_1) & & k(x_N,x_N) \end{bmatrix}.$$

The formulation (3.13) corresponds to a nonlinear least-squares problem in view of the residual vector $r = (r_1, \cdots, r_{M_z})$, where each residual is defined by

$$r_j(x;\theta) = h(x, f(x;\theta)) - z_j, \quad j = 1, \ldots, M_z.$$

The gradient of each residual can be readily computed as

$$\nabla_\theta r_j(x,\theta) = K^{-1}\bar{k}(x)\nabla_y h(x, f(x)).$$

A desirable property of the gradient $\nabla r(\theta)$ is that $\nabla r(\theta)\nabla r(\theta)^T$ is full rank, which requires that $M_z \geq M_y n_y$. Note that all quantities $K^{-1}\bar{k}(x_j)$ are precomputed and cached before a least-squares solver (such as Levenberg-Marquardt) is applied.

The resulting error covariance of the estimate θ^* can be approximated as

$$P \approx \frac{1}{M_z - M_y n_y} \|r(\theta)\|^2 (\nabla r(\theta) \nabla r(\theta)^T)^{-1}.$$

in case of uninformative priors (with $P_0 = \infty$). This expression can be easily extended to the case when a prior is employed.

Experiment

The proposed latent GP approach is now applied to the palpation problem. The measurement $z \in \mathbb{R}_{\geq 0}$ denotes the norm of a force recorded by the robot at a commanded end-effector point $p = (p_x, p_y, p_z) \in \mathbb{R}^3$. Our model has two hidden outputs $y = (\kappa, s)$ where κ denotes the surface stiffness and s denotes the surface height over a given point $x = (p_x, p_y)$. The function h then defines the physical relationship between force, depth, and stiffness according to

$$h(x, y) = \kappa(s - p_z(x)),$$

where the function $p_z(x)$ simply looks up the recorded height at given recorded location $x = (p_x, p_y)$. Force measurements are assumed to have variance $\Sigma_\epsilon = \sigma_f^2$ with $\sigma_f = 0.01$ N.

The two outputs are modeled as independent GPs in which case diagonal kernel

covariance becomes

$$k(x, x') = \begin{bmatrix} k_\kappa(x, x') & 0 \\ 0 & k_s(x, x') \end{bmatrix},$$

where k_κ defines the GP covariance for the stiffness and k_s gives the GP covariance for the surface. A standard squared exponential kernel is employed for both with different parameters.

The gradient of the force required for the solution of (3.13) is simply

$$\nabla_y h(x, y) = \begin{bmatrix} s - p_z(x) \\ \kappa \end{bmatrix}.$$

This approach was applied to the palpation problem using $M_z = 2000$ data points by selecting the locations x_j in the latent GP dataset D_y to be on a regular grid with 5mm cell size, resulting in $M_y = 256$ points. For this experiment, very wide priors were employed, specifying that the surface mean is at 15 mm with a standard deviation of 20 mm and a stiffness at 500 N/m with a standard deviation of 1000 N/m were used. Figure 3.7 compares the resulting estimated function to the ground truth estimates from 77000 points.

Although the accuracy of latent GP estimation was comparable to the direct GP, this approach was not explored further as the former direct GP estimation technique was extensible to fit in the CSA framework and was more efficient to implement.

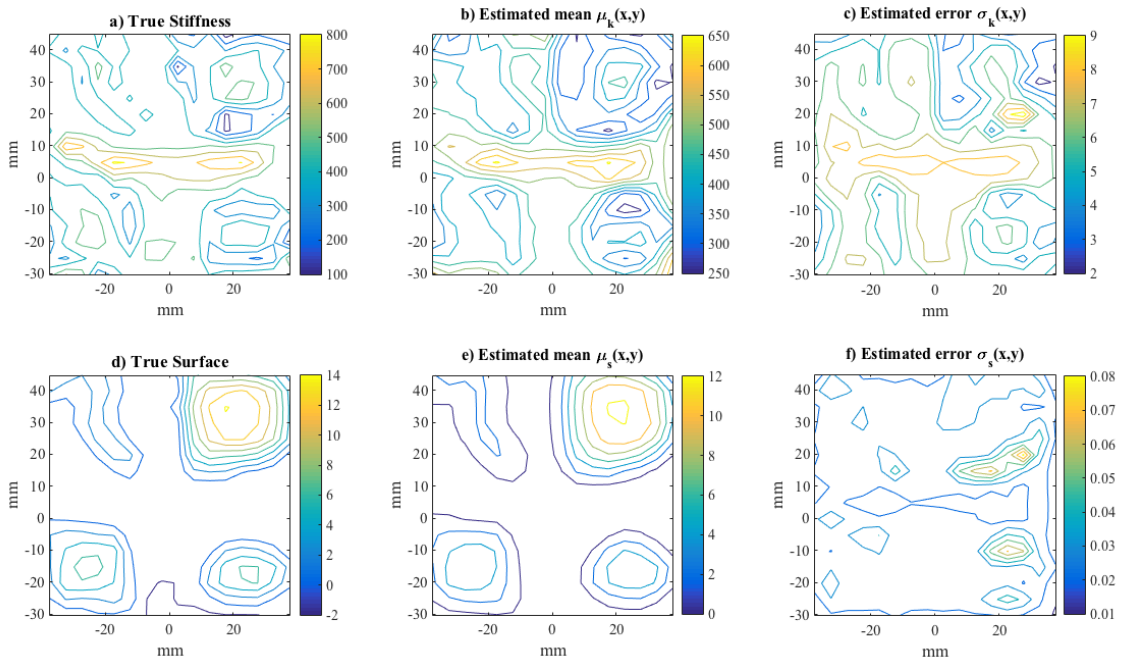


Figure 3.7: Latent GP estimation a) Contour plot of true stiffness; b) Contour plot of estimated mean of stiffness; c) Contour plot of estimated error in stiffness; d) Contour plot of true surface height; e) Contour plot of estimated mean of the surface geometry; f) Contour plot of estimated error in surface geometry

3.3.4 Online GP Estimation

In the previous subsection (3.3.2), offline estimation techniques independent of palpation strategy were presented to estimate organ shape and stiffness using GP. It demonstrated the feasibility of concurrent estimation of the surface geometry and stiffness from continuous palpation trajectories. However, the execution time increased quadratically with the amount of training data, thus, defying the real-time claim of CSA. One method to speed up the estimation would be to sample the incoming data at a much lower rate. However, this would result in the loss of valuable training data and such subsampled data can still introduce numerical instabilities when the exploration trajectory crosses over itself.

To overcome these limitations, we developed a technique to improvise the direct GP estimation to provide reasonable support for GP, while using small sets of training data to expedite the estimation process. This allows having near-video frame rate updates of the organ geometry and stiffness. One of the key challenges in providing such "online" updates is data redundancy. GP estimation is extremely sensitive to redundant data and its computational efficiency is affected by the amount of data used in training. To tackle this problem, a spatial data structure was used to prune redundant data and to efficiently retrieve data for training purposes. Figure 3.8 shows a conceptual illustration of both the offline direct GP technique and the online technique. In the offline technique, the estimation of geometry and stiffness is predicted using a training model over all the data. On the contrary, the online technique, dis-

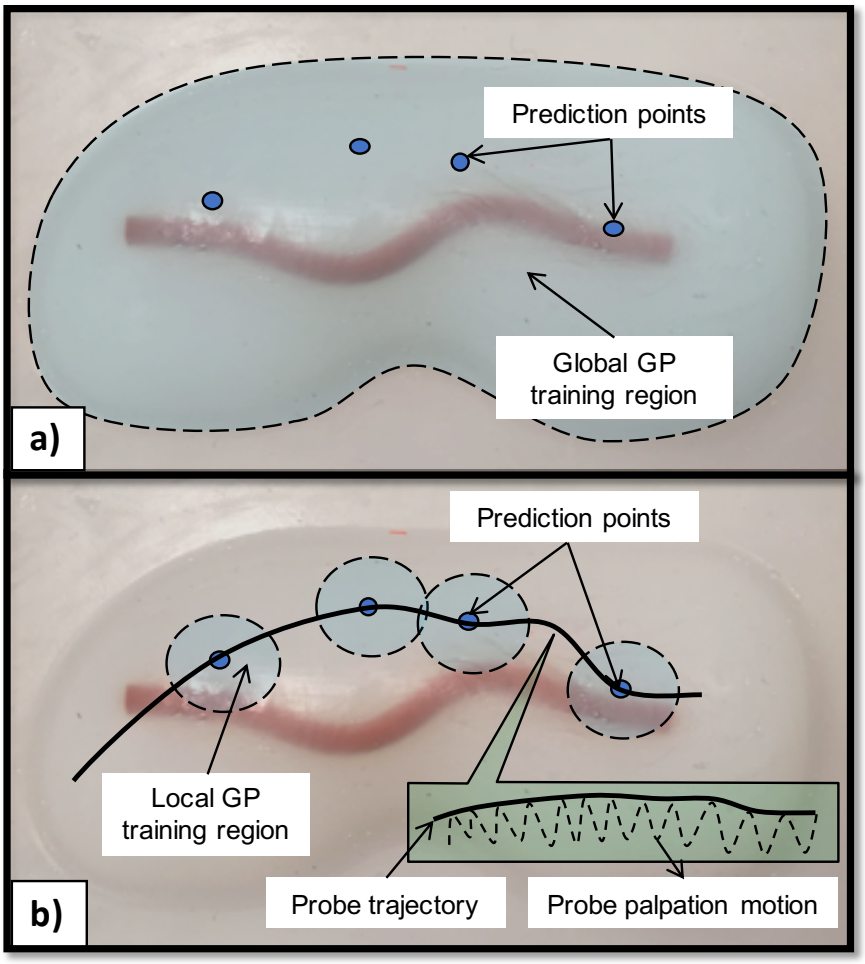


Figure 3.8: Conceptual representation of offline and online estimations of stiffness and surface geometry: a) offline technique requiring training a GP over all the data, b) online surface and stiffness estimation using only nearby information to estimate surface information along the palpation trajectory

cussed in this section, only uses the neighboring information to train the GP model. The proposed online estimation technique uses the core formulation of GP with the main focus on delivering near-video frame rate updates of the organ geometry and tissue stiffness during force controlled exploration/palpation.

Hash grids

A brief overview of spatial data structures, in particular, hash grids, is provided here for understandability of the reader. Spatial hashing[102,103] can help speed up certain operations like collision, ray-tracing, nearest neighbor search, etc. For the online estimation, the most important capability is acquiring local/neighborhood information around a given region of interest in constant time. While there are other advanced data structures for spatial indexing, hash grids were chosen because of their straightforward implementation and simple structure. A hash grid (HG) is a two or three-dimensional extension of a hash table. It's a simple data structure that subdivides the 2D/3D space into uniformly-sized 'bins' or 'buckets'. The basic idea of a hash table is that a piece of data (the 'key') is passed through some function (hash function) to generate a new value (the 'hash') which is used as an index into a set of buckets. Figure 3.9 shows a visual representation of a 3D hash grid, where the red cuboids represent a bucket/bin and the blue dots represent 3D points.

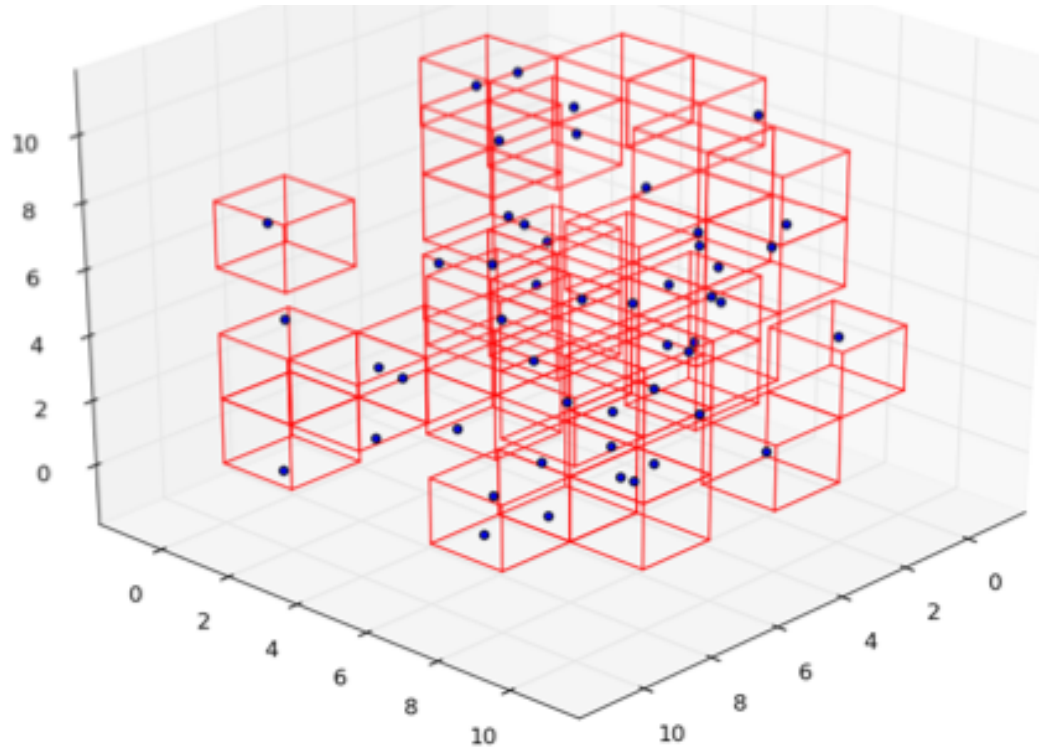


Figure 3.9: 3D spatial grid (Image taken from http://www.sgh1.net/)

Notations

A few important notations that are used in the following subsection are listed in Table 3.1 for the reader's reference.

HG_t	Training hash grid
HG_p	Prediction hash grid
X	All position data stored in HG_t
Y	All force data stored in HG_t
$\mathcal{T}_{gp}$	$(X_s \ll X, Y_s \ll Y)$ position-force pairs used for training
$\mathcal{P}_{gp}$	Set of query points for prediction
p_c	Current tip position
τ_c	Sensed forces
p_q	Query point
$h(p)$	Hash ID for point p
$p_q(h)$	Query point stored in the bucket with hash ID h
$\mathcal{N}_t(h)$	Neighboring hash IDs of h in HG_t
$\mathcal{N}_p(h)$	Neighboring hash IDs of h in HG_p

Table 3.1: Variables used in the proposed online GP estimation technique

The Proposed Online Estimation Methodoloy

In the offline stiffness estimation, all the position-force pairs were used to train the GP model. This results in a predictive distribution of force over the entire surface. However, based on the model assumptions mentioned in Section 1.2, not all the training data is needed to estimate a predictive force distribution with acceptable accuracy at a certain location. Suppose that we have a GP model over forces based on position data X and its corresponding force data $\mathcal{T}$. Now, given a query point p_q, a predicted mean and variance can be obtained according to (3.11) and (3.12). Since it is assumed that the deformations from palpation are local, the position-force pairs far away from the query point will have a negligible contribution to the predicted distribution at p_q. Thus, a subset of position-force pairs $X_s << X$ and $\mathcal{T}_s << \mathcal{T}$ are sufficient to predict the force distribution at p_q. In order to implement the proposed online estimation method, two continuous tasks were developed using the *cisst* component-based framework;

(a) *mtsGPCollector*: This continuous task reads the contact locations of the tip of the slave robot with the surface of the tissue, and stores them in a hash grid. Further, it estimates the query point based on the current tip location and packs the query point along with the required training data for estimation.

(b) *mtsGPEstimator* : This continuous task retrieves the training set from the *mtsGPCollector* and performs a direct GP estimation as described in Section

3.3.2. The estimated data is sent back to *mtsGPCollector* and is stored in a different hash grid.

Both these processes are executed in parallel to make sure one component always collects and stores the data, and the other is only used for GP training and prediction. A brief overview of component-level communication is shown in Figure 3.10. The provided interface of the *mtsGPCollector* is connected to the required interface of the *mtsGPEstimator*. This allows the *mtsGPEstimator* to retrieve the latest training and query sets and also to update the prediction grid upon completion of the prediction procedure. This workflow consists of three major parts,

(a) Add/retrieve data from training grid

(b) Select when/where to predict, i.e., how to calculate the query point.

(c) Train position-force data, and Predict geometry and stiffness.

A) Add/retrieve data from training grid

This subprocess is executed by the *mtsGPCollector*. For fast retrieval of $\mathcal{T}_{gp} = \{X_s, \mathcal{T}_s\}$, the training grid represented by HG_t is used. HG_t is preallocated with empty buckets based on a user-specified grid size ($gSize$) and min-max limits ($minLim, maxLim$) in each dimension of the grid. The grid size should be specified based on the feature that is being identified. For example, if we are trying to locate an artery-like feature, then the grid size should be the approximate diameter of the artery. The grid size,

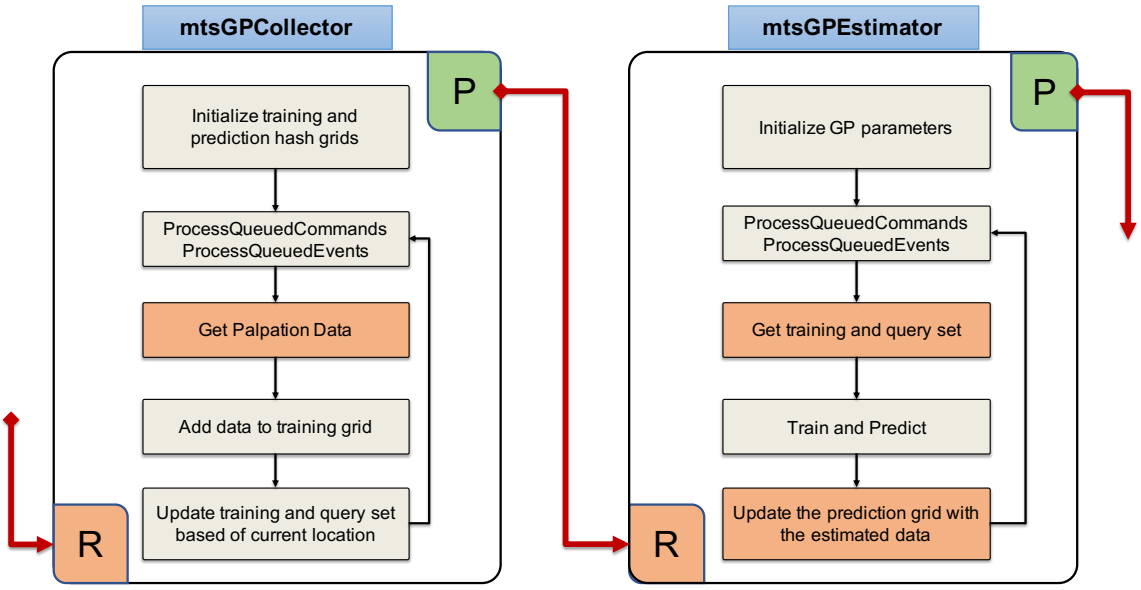


Figure 3.10: GP collector and estimator components. 'P' and 'R' represent provided and required interfaces, respectively. Orange colored boxes represent access to data/-function via interface communication. (More details about interface connections are described in Section 2.3).

gSize, is chosen such that, the data that are $3 * gSize$ units away from each other have a minimal correlation. This can be assumed based on the model assumptions described in Section 1.2. Any given bucket, with hash ID h, in HG_t contains the following information;

- A FIFO (first-in-first-out) queue of position-force vector pairs, $\mathcal{Q}(h)$. Since the data is collected at a very high frequency (usually the frequency at which the robot is controlled), queue size should be large enough to store a lot of information. However, during the retrieval process, sparse amount of data is randomly sampled.

- A precomputed vector containing hash IDs of neighboring buckets, $\mathcal{N}_t(h)$. Information of the neighboring bucket is calculated based on linear indexing of grid cells.

When a new position-force pair is obtained, the following steps take place in a sequential order,

- **Calculate hash ID :** Given a palpation point $p_c = [p_c^x, p_c^y, p_c^z]^\top$ and corresponding force τ_c, the hash function used to generate the bucket/hash ID is

$$h(p_c) = i + gSize.X * j + gSize.X * gSize.Y * k \tag{3.14}$$

where,

$$\begin{aligned} i &= \frac{floor(p_c^x - minLim)}{BucketSize.X} \quad , \quad BucketSize.X = \frac{maxLim.X - minLim.X}{gSize.X} \\ j &= \frac{floor(p_c^y - minLim)}{BucketSize.Y} \quad , \quad BucketSize.Y = \frac{maxLim.Y - minLim.Y}{gSize.Y} \\ k &= \frac{floor(p_c^z - minLim)}{BucketSize.Z} \quad , \quad BucketSize.Z = \frac{maxLim.Z - minLim.Z}{gSize.Z} \end{aligned}$$

$\{p_c, \tau_c\}$ is then added to the FIFO queue $\mathcal{Q}(h(p_c))$.

- **Package training data :** If a query point p_q is selected, an appropriate set of X_s and $\mathcal{T}_s$ values needs to be retrieved from HG_t. To retrieve this data from the training grid, first, the corresponding hash ID $h(p_q)$ is calculated using (3.14). Then, the iteration over all the neighboring hash IDs stored in $\mathcal{N}_t(h(p_q))$ is done to retrieve a certain number of random position-force pairs from their respective queues. An optimal bucket size is empirically calculated such that a minimal number of buckets are sufficient to characterize the local surface geometry and tissue stiffness.

  This process is computationally inexpensive since at any time the maximum number of neighbors in 2D and 3D grids would be 8 and 26, respectively. Once we have $\mathcal{T}_{gp}$, we can proceed to train and generate a GP model of forces in the neighborhood of p_q. The pseudocode for retrieving the training data set for GP is given in Algorithm 3.2.

Algorithm 3.2 Retrieving Training Data

Input: HG_t, p_q
Output: $\mathcal{T}_{gp}$
`hId = h(`p_q`)` ▷ Hash ID of the query point
$\{\mathcal{T}_{gp}\} = \mathcal{Q}$(`hId`) ▷ Initialize the queue with data from hId
for `each h in` $\mathcal{N}_t$(`hId`) **do**
$\mathcal{T}_{gp} \leftarrow \mathcal{T}_{gp} \cup \mathcal{Q}(h)$ ▷ Add data from neighbors of hId to the training queue
end for

B) Query point selection

This subprocess is executed in the *mtsGPEstimator* component while some commands are called from *mtsGPEstimator* upon completion of the prediction procedure, via interface communication. Query point selection is one of the main challenges in estimating the surface geometry and stiffness since the selection criteria will impact the accuracy of estimation. As mentioned earlier, a query point should be deep enough inside the surface where the force vector is noise-free. The information is being gathered during a palpation task (described later in Section 5.1.3), where the desired force applied by the palpation tool on the tissue is set to follow a sinusoidal profile. This allows storing the contact point as a query point when the force is maximum. For reference, following is the force motion (F_d^*) commanded to the slave tip for palpation (complete details are in Section 5.1.3),

$$F_d^* = F_d + A * Sin(2\pi\omega t)$$

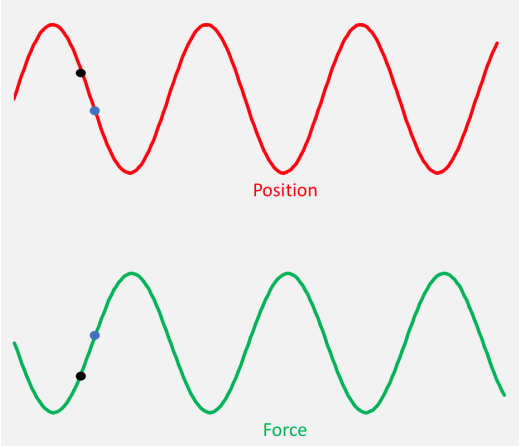
where A, ω, t are parameters of the sine motion. Here, the force will be maximum when the sine function is 1, i.e, $Sin(2\pi\omega t) = 1$. However, due to delay in the low-level controller, the time at which the maximum force is actually applied by the probe tip will lag slightly from the time at which the force is commanded (Figure 3.11). Thus, adding the position-force pair when the sine value reaches 1 is not appropriate. Instead, when the algorithm asks for maximum force, from that instant a search is initiated to find the contact position, p_c, at which the force magnitude starts decreasing. Thus, a query point $p_q = p_c$ is added to HG_p when the palpation force τ_c satisfies the condition described in Algorithm 3.3. Each point p_q added to HG_p now inherently corresponds to a p_{far} stated in (3.11).

Algorithm 3.3 Adding to the Prediction Grid

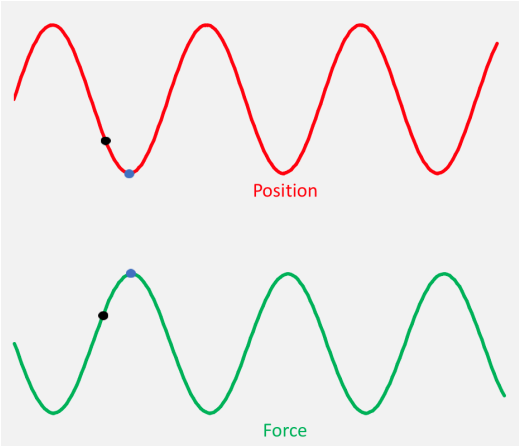
Input: p_c,τ_c,maxF,fSearch
Output: HG_p,maxF,fSearch

```
hId = h(p_c)
if value of sine function = 1 then
    fSearch = true                                  ▷ Start searching
    maxF = 0
end if
if fSearch then
    if τ_c > maxF then
        maxF = τ_c                                  ▷ Force is still increasing
    else
        p_q(hId) = p_c                              ▷ Add query point to HG_p
        maxF=∞, fSearch=false                       ▷ Reset
    end if
end if
```

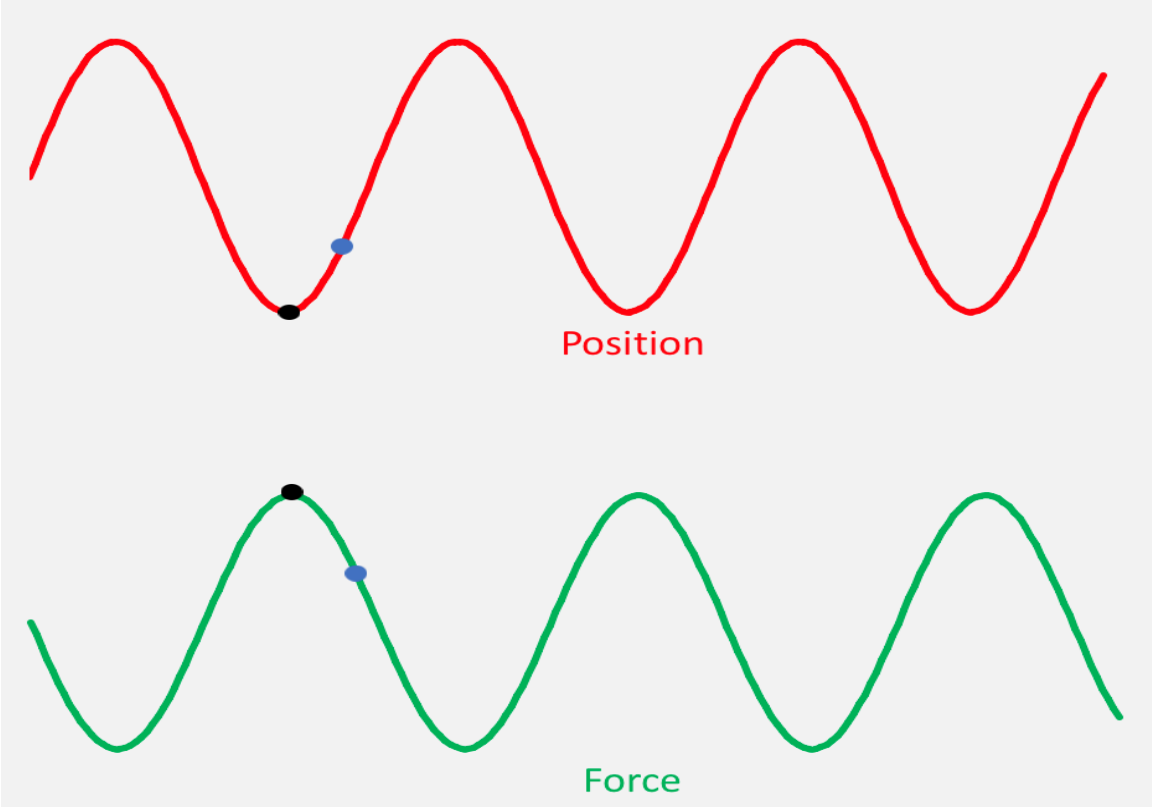
Note that all the palpation points are not stored in the prediction grid, but only



(a) Control lag between commanded and actual motion/force



(b) Commanded force is maximum; start search, *i.e.* `fSearch = true`



(c) Sensed force is maximum at this state, when the sensed force starts decreasing

Figure 3.11: Sinusoidal motion in force profile. Blue dot corresponds to commanded force/motion and black dot corresponds to sensed force or current location.

the ones that are deep inside the tissue, where the signal-to-noise ratio is higher, are identified and stored. Now, to retrieve the query point based on the current contact point p_c, one just needs to retrieve the query point stored in the prediction grid HG_p based on the spatial location of p_c. Algorithm 3.4 describes the process of retrieving the set of query points, $\mathcal{P}_{gp}$, for prediction purposes, which includes p_q and its neighbors.

Algorithm 3.4 Retrieve Query Set

Input: $\mathcal{P}_{gp}$
Output: p_q, $\mathcal{N}_p(p_q)$

```
hId = h(p_c)                          ▷ Calculate hash ID for p_c
p_q = HG_p(hId)                       ▷ Get the query point
P_gp = p_q                            ▷ Add p_q to set of query points
for each h in N_p(hId) do
    P_gp = P_gp ∪ p_q(h)
end for
```

C) Train position-force data, and Predict geometry and stiffness

This subprocess is executed in the *mtsGPEstimator* component to train the GP model using the training data received from the *mtsGPCollector* component via interface communication. Query points $\mathcal{P}_{gp}$ are also retrieved using the same interface communication for prediction purposes. Using the formulation described in Section 3.3.2 mean estimates of the organ geometry and tissue stiffness along with their uncertainty are calculated at all the retrieved query points using the trained model, $\mathcal{GP}$. The pseudocode for training and prediction procedures is detailed in Algorithm 3.5.

Algorithm 3.5 GP Train and Predict

Input: $\mathcal{T}_{gp}$, HG_p, p_q
Output: Updated HG_p

```
hId = h(p_q)
Calculate μ_κ^hId(p_q), V_κ^hId(p_q)        ▷ Calculate mean stiffness and variance
Calculate μ_s^hId(p_q), V_s^hId(p_q)        ▷ Calculate mean surface point and variance
for each p in P_gp do
    Update μ_κ^h(p), V_κ^h(p)
    Update μ_s^h(p), V_s^h(p)               ▷ Update neighboring information of p_q in HG_t
end for
```

Overall communication flow

Figure 3.12 shows the entire communication details for the online estimation strategy. Once a new palpation point-force pair, (p_c, τ_c), is obtained from the robot, it is added to the HG_t using the formulation in Section 3.3.4. Based on the current position p_c, query set $\mathcal{P}_{gp}$ is retrieved from HG_p using Algorithm 3.4, and training data $\mathcal{T}_{gp}$ is retrieved from HG_t using Algorithm 3.2. Simultaneously, the palpation point p_c is also added to HG_p based on the decision process described in Algorithm 3.3. A $\mathcal{GP}$ model is trained using the training set $\mathcal{T}_{gp}$ followed by an estimation procedure using the query set $\mathcal{P}_{gp}$, as described in Algorithm 3.5. Updated estimation information is added to HG_t for later storage and display purposes.

Experiment

In this subsection, we present the results of the online estimation of tissue stiffness and surface geometry during teleoperation.

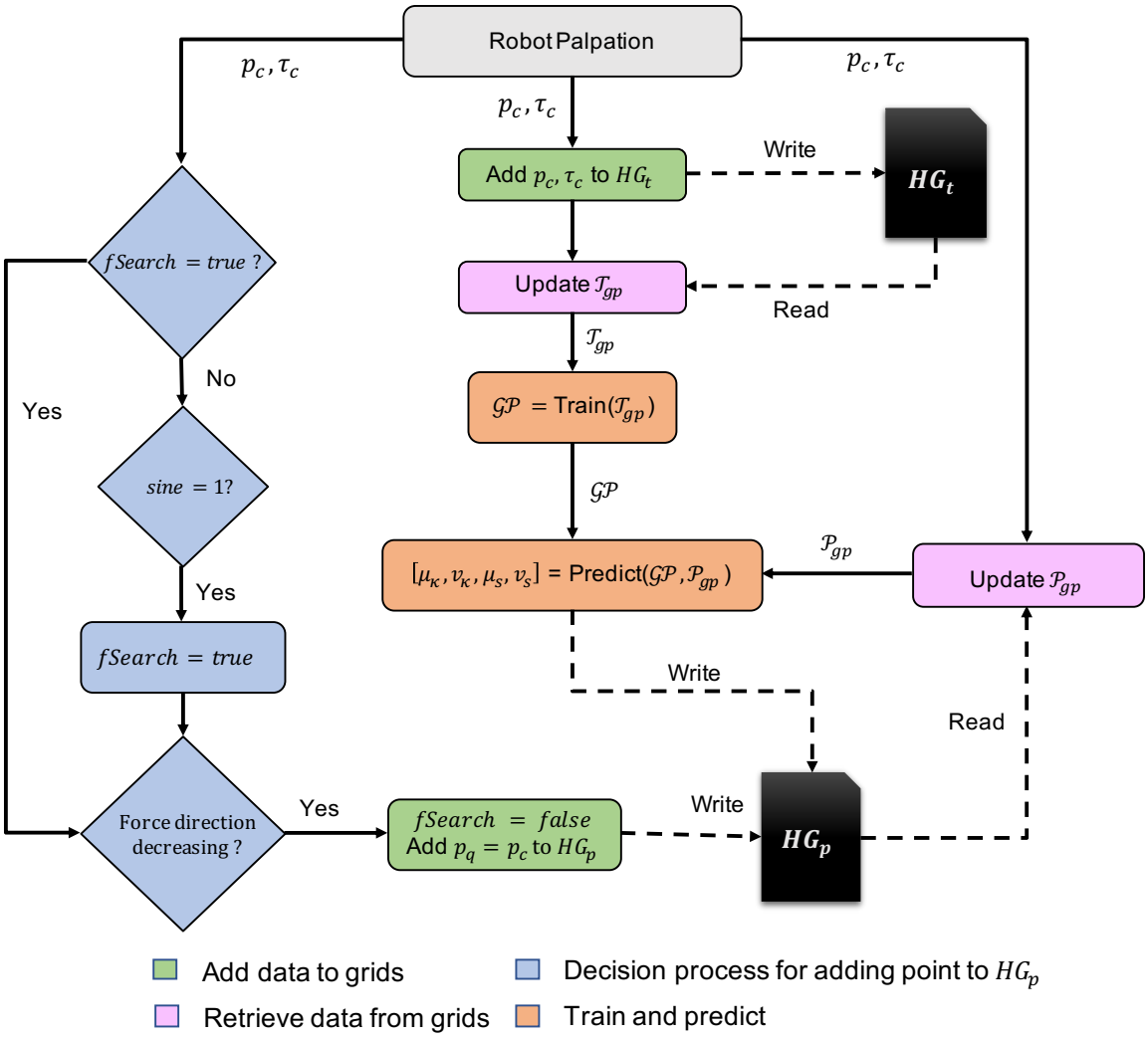


Figure 3.12: Communication flow of online geometry and stiffness estimation

We used a simulated surgical scenario to demonstrate the feasibility of using our online estimation in an interactive surgical assistant. Our scenario assumes that the user has an intuition about the approximate location of the stiff inclusion, comparable to what might be obtained from a segmented CT or MRI image. For this experiment, we chose a different phantom, shown in Figure 3.13, which is softer than the phantom used in the previous experiment (Figure 3.15).

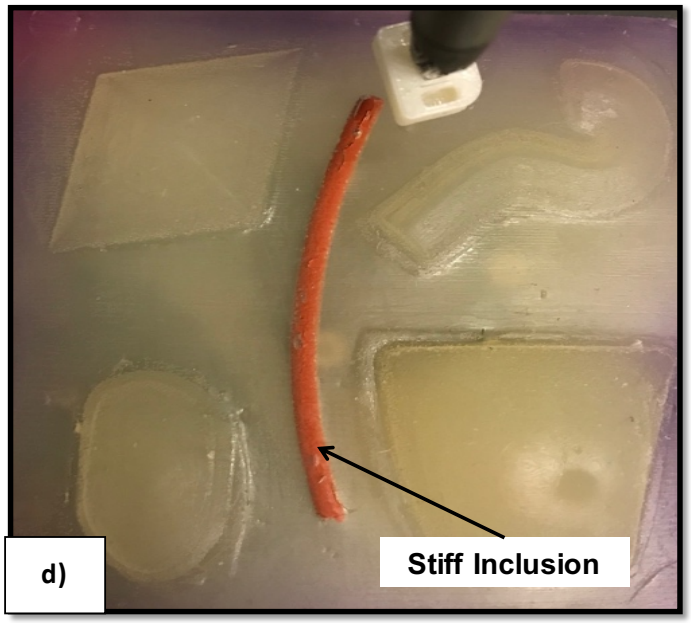


Figure 3.13: Soft phantom (100 mm x 100 mm) with an embedded rubber wire resembling an artery

We used the method described in Section 3.3.5 to generate ground truth for this phantom. Since the surface of the model is soft, exerting large forces will puncture the tissue, therefore, the force controller reference was set at 0.3 N normal to the surface. Additionally, this allowed us to demonstrate the working of our algorithm where the force normals are a bit noisy due to low range forces. Approximate registration was

performed by pointing to the corners of the phantom, and a user-defined region of interest (ROI) was acquired for exploration, as explained in Section 6.1. A virtual fixture was defined using the framework defined in [104, 105] to constrain the motion of the probe to the interior of the region. The user teleoperated the probe along the surface of the anatomy, while the algorithm superimposed a palpation motion based on a sinusoidal reference force in the direction normal to the surface (Section 5.1.3). We call this semi-autonomous teleoperation. The stiffness map was continually updated during palpation.

This exploration strategy closely resembles a surgical setup where the surgeon would like to know the stiffness of an underlying tissue while palpating. Our online estimation allows this capability by providing near-video frame updates of the stiffness map which would help the surgeon make an informed decision on the trajectory of teleoperation. While the user was teleoperating, our online estimator received data at 200 Hz and the surface stiffness was updated at 50 Hz. Figure 3.14c shows the stiffness map generated by the online estimation after teleoperated exploration of the ROI.

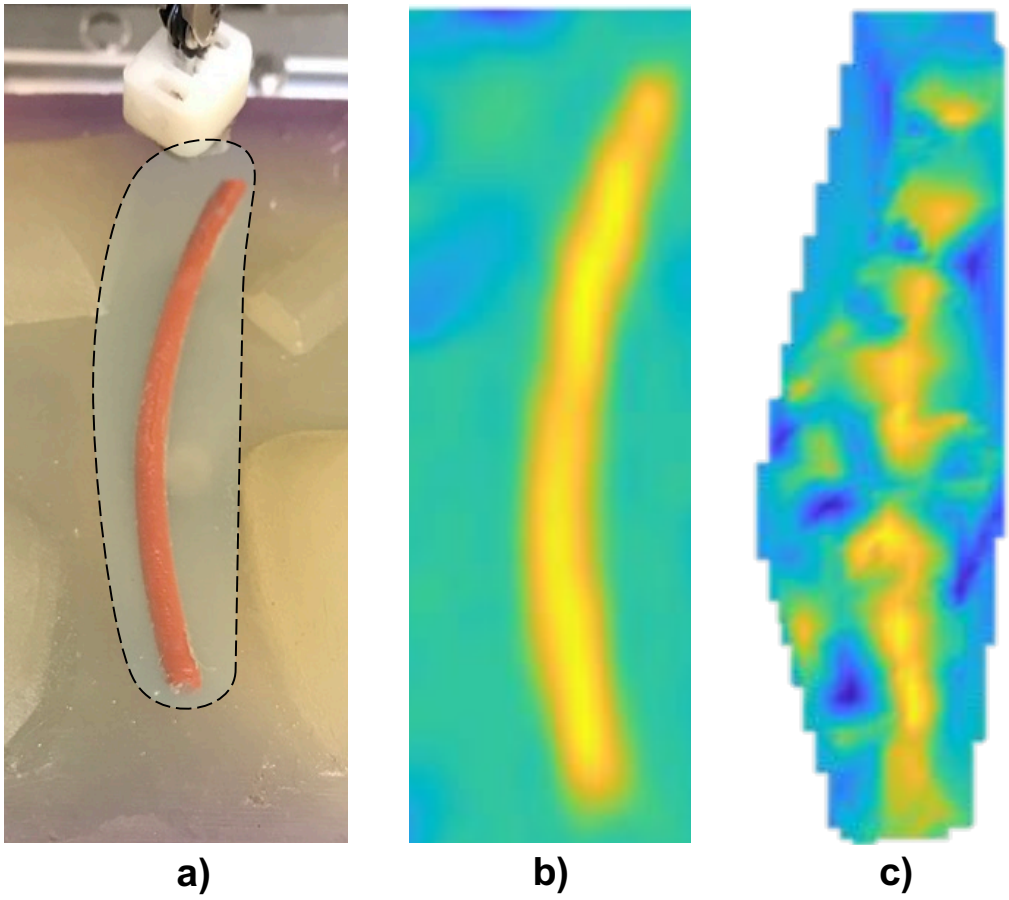


Figure 3.14: Semi-automated teleoperation experiment with our online estimator receiving data at 200 Hz. a) User-specified region of interest for semi-autonomous teleoperation, b) Ground truth stiffness map, c) Estimated stiffness using the online method

3.3.5 Comparison and Evaluation of Online Estimation

In this subsection, the efficiency of the online strategy is compared with the direct GP offline approach described in Section 3.3.2. Further results of the online strategy are evaluated for stiffness estimation. In this application, dVRK was used with an 8mm needle driver for teleoperation. Since the dVRK manipulators do not have built-in force sensors, an ATI Mini 40 Force-Torque sensor was attached below the phantom. Other methods (*e.g* [63]) can be used to estimate tool-tissue interaction forces, so the results are not limited to this particular setup. This experiment was performed on an Intel(R) Xeon(R) CPU E5-1620 v2 @ 3.70GHz with 24 GB RAM. The phantom used for this experiment was made of silicone elastomer (M-F Manufacturing, Fort Worth, Texas), resembling the shape of a kidney. The phantom contained a stiff rubber inclusion resembling a subsurface artery. [i]

Comparison

Autonomous palpation was performed on a silicone phantom (Figure 3.15) with an embedded rubber wire resembling an artery. Optimum palpation frequency of 2 Hz and amplitude of 1 N were empirically calculated for minimal harmonic excitation in the force sensor while palpating. The mean desired force of 1.5 N was chosen to

[i] This experiment was performed at Johns Hopkins University with remote assistance from Vanderbilt University; Members include Long Wang, Rashid Yasin, Nabil Simaan, and Russell H. Taylor. Details of this experiment can be found in [104].

ensure that the probe tip remains in contact with the phantom model at all times. The probe was then given a preplanned trajectory on the phantom for continuous palpation.

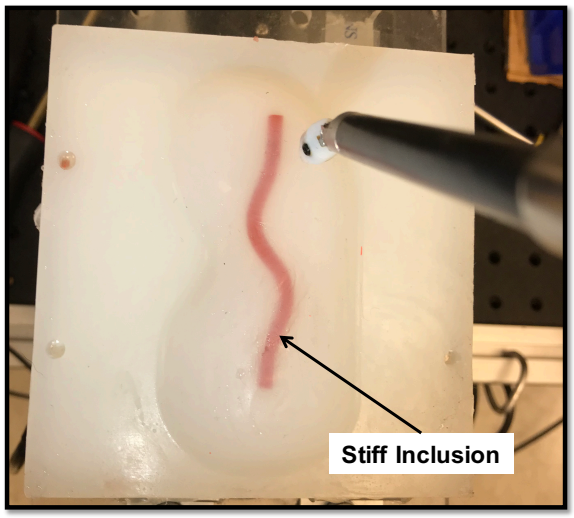


Figure 3.15: Stiff phantom (100 mm x 100 mm) with an embedded rubber wire resembling an artery

To compare the efficiency and accuracy of the offline and the online estimation methods, we set up two estimator blocks which received this palpation data at 200 Hz and updated the stiffness map as fast as possible. We continuously streamed the estimated surface information to MATLAB for display purposes. The first estimation block was based on the offline method, in which the estimation is performed recursively using all the data currently stored until that iteration. The second estimation was based on the online approach, in which the estimation is performed while the user is palpating. For comparison purposes, we used the same values for GP hyperparameters in both cases. The length scale (l) was set to be 6 mm and variance (σ_f) was

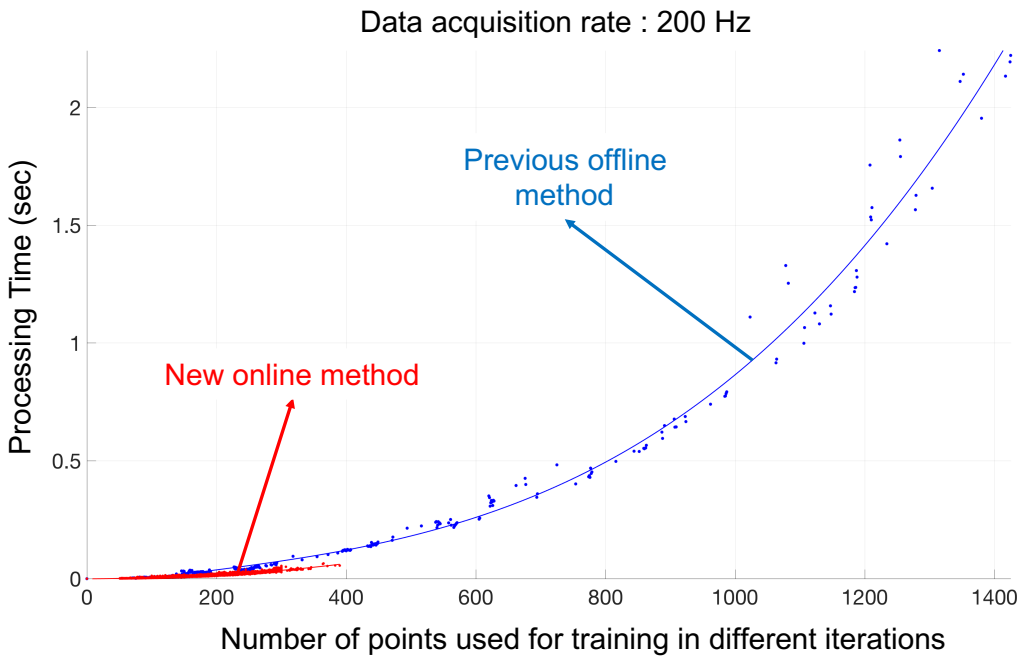


Figure 3.16: Time taken to train the GP model and predict stiffness at each iteration for the first 1425 sampled training data (position-force) pairs.

set to be 0.9 N. In Figure 3.16 we can clearly see that the processing time for offline estimation increases quadratically with incoming data, since all the data is trained and the estimation is done on the region palpated till that iteration. Eventually, the offline method took $\approx$ 9.5 seconds when the number of training data pairs reached 2400. However, the online estimation trains a maximum of 380-400 data pairs, resulting in an average processing time of 0.02 sec (50 Hz) in each iteration. Use of the fast spatial data structure enables these points to be chosen appropriately for training the GP model without excessive local redundancy in the data used. Additionally, the online method only requires neighboring information for local stiffness update, thus limiting the number of training data pairs in each iteration.

Evaluation

To evaluate the accuracy of the online algorithm for stiffness estimation, we compared our results with the ground truth surface and stiffness information. True surface information was calculated by deploying a discrete probing strategy using a high stiffness Cartesian stage robot. To generate a dense distribution of probing points in a grid format with 1 mm spacing, the robot recorded 10 measurements at each probe location by probing along the normal up to a depth of 3 mm in increments of 0.3 mm. The surface point was computed by using a regression fit on forces and z-positions along the normal. A simple linear model was used to calculate the ground truth stiffness at that location using two position-force pairs along the normal. Figure 3.17a shows the ground truth information generated. In this experiment, data was received at a frequency of 200 Hz for both the offline and the online method. In the case of the offline method, the incoming data was stored and the stiffness map was estimated after trajectory completion as shown in Figure 3.17b. In the case of the online estimation technique, the stiffness map was updated at a frequency of 47 Hz and the end result is shown in Figure 3.17c.

As shown in Table 3.2, our online method estimated the surface geometry and stiffness with at least comparable accuracy to the previous offline method. Our experience with previous work[106] and our conversations with surgeons indicates that the accuracy reported by the online estimation is sufficient for a surgeon to detect stiff features relative to adjacent softer tissues.

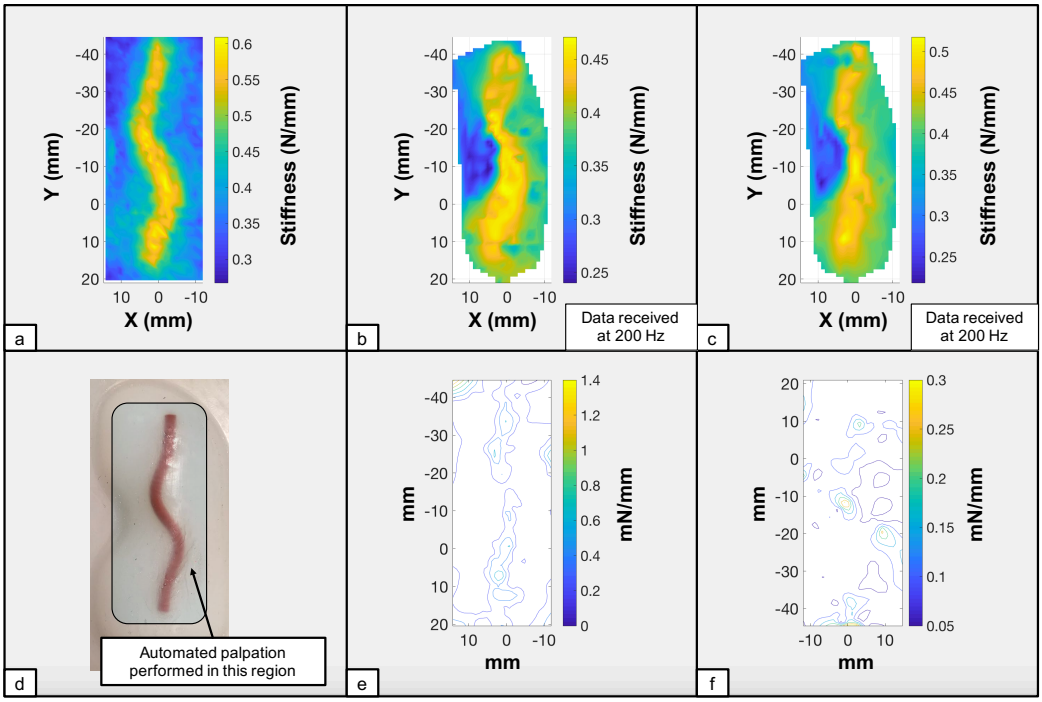


Figure 3.17: a) Ground truth stiffness, b) Stiffness estimation using offline method, c) Stiffness estimation using online method, d) Silicone model with highlighted region used for palpation, e) Contour plot of the estimated stiffness variance using offline method, f) Contour plot of stiffness variance using online method.

	Offline estimation	Online estimation
geometry RMS error (mm)	1.3676	1.3203
stiffness RMS error (mN/mm)	80.4	69.8

Table 3.2: RMS errors for stiffness and geometry based on previous offline estimation[106] and current online estimation using automated palpation.

Repeatability

We also performed multiple trials with different data acquisition rates. The palpation setup remained the same as described above, however, the rate at which the online estimator receives the position-force data is varied. We compared the results from each trial with the ground truth stiffness and geometry data and the RMS errors for the same are detailed in Table 3.3. As we can see from the table and Figure 3.18, our online estimation produced comparable results across a wide range of sample rates.

frequency (Hz)	geometry RMS error (mm)	stiffness RMS error (mN/mm)	update rate (Hz)
50	1.3541	70.4	43
100	1.3729	72.5	42
200	1.3514	69.5	45
400	1.3446	70.3	50
500	1.3246	70.3	52
1000	1.3701	77.3	53

Table 3.3: Geometry and stiffness results for different sample rates using the online estimation method.

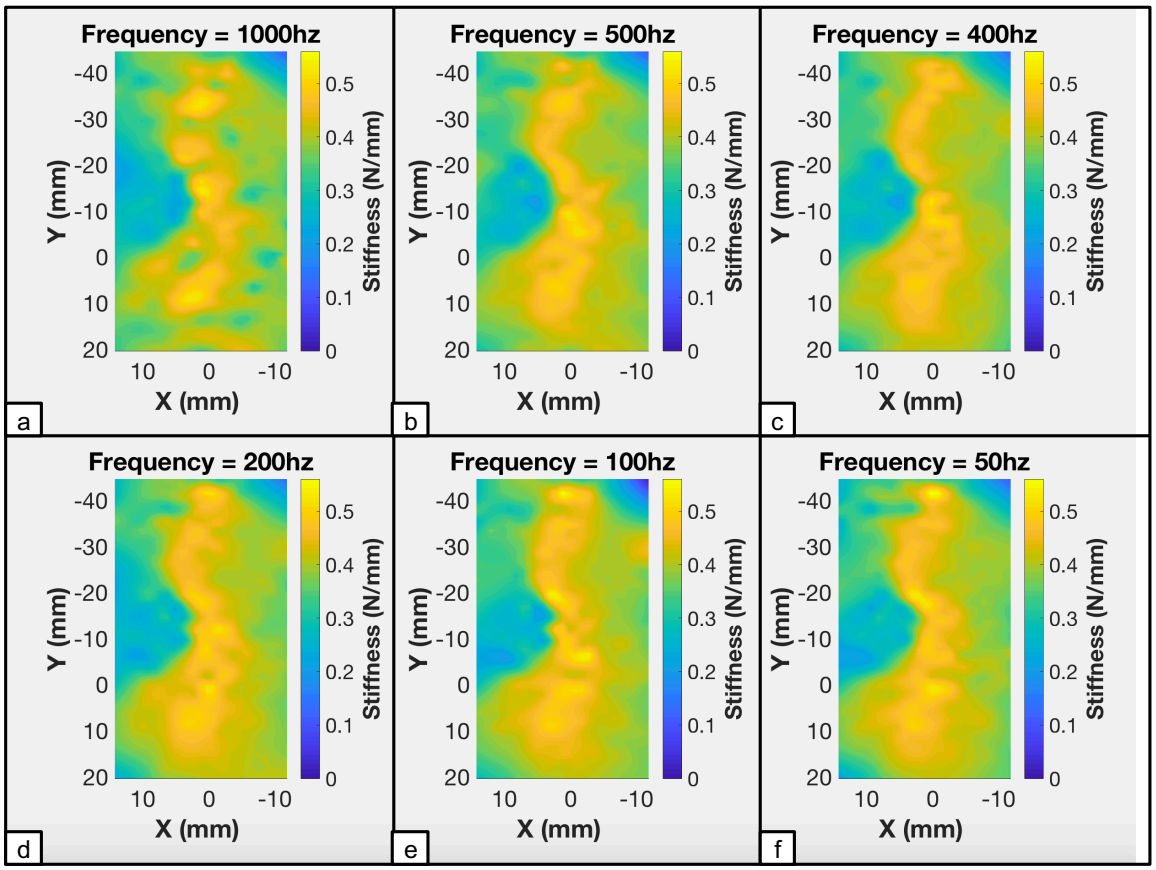
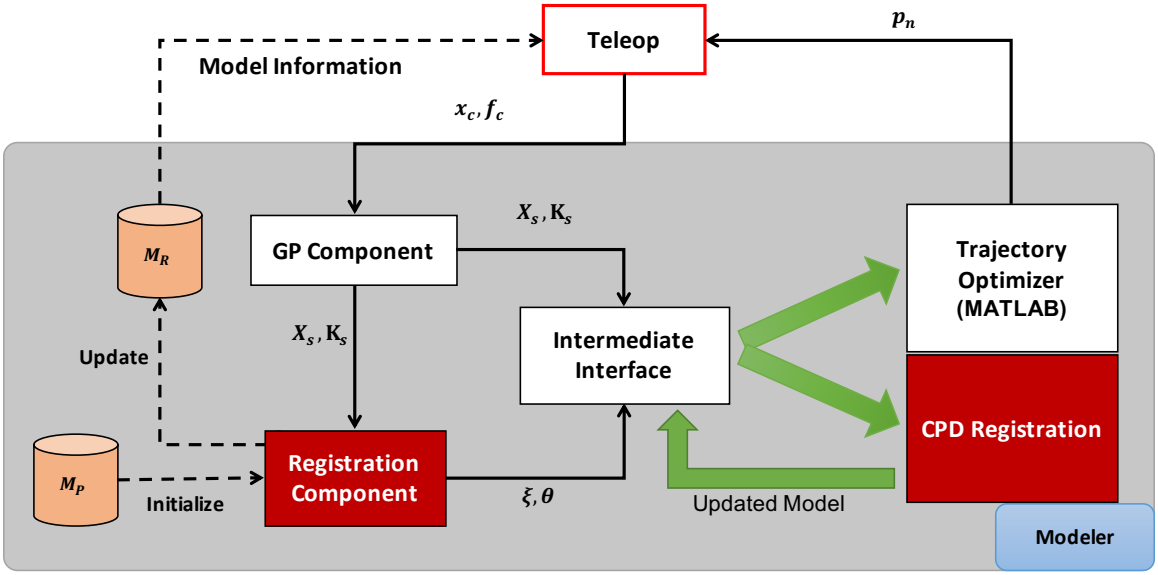


Figure 3.18: Stiffness maps generated by online estimation at various data acquisition rates.

3.4 Model Registration



Model registration is performed in two steps: 1) a rigid registration based on Iterative Most Likely Point (IMLP), and 2) non-rigid registration based on Coherent Point Drift (CPD). Both these subprocesses together are responsible for providing the latest model information by registering the current model with the information provided by the GP component. In the following subsections, necessary data storage required for registration is discussed, followed by a brief overview of IMLP and CPD. The intermediate interface used to send the output information of IMLP to CPD (in MATLAB) is also detailed at the end of this section.

3.4.1 Registration Algorithms

Iterative Most Likely Point (IMLP)

The IMLP algorithm, developed by Billings et al.,[96] is a robust probabilistic algorithm for registering positional feature data that is characterized by anisotropic uncertainty. In order to converge towards the correct solution, within each iteration, the algorithm dynamically adapts its noise model. It further uses the noise model to detect and mitigate outliers. As discussed in [96], an efficient IMLP algorithm consists of two phases :

1. **Correspondence Phase :** In this phase, the most likely match from the model shape is computed in an efficient manner. A variant of the K-Dimensional (KD) tree called the Principal Direction (PD) tree[107] (also known as the covariance tree), is used to search for an optimal point of correspondence on the model shape. Both KD and PD trees partition the geometric space into a hierarchy of nodes, however, the primary variation lies in the orientation of the coordinate systems. The PD tree has a local coordinate system that is oriented based on the geometric dispersion of the resident data, unlike the KD tree, where the local coordinate system is axis-aligned with the global coordinate system.

2. **Registration Phase :** In this phase, a linear least-squares solver is set up to minimize the match error between the model shape and the corresponding point sets that are characterized by anisotropic uncertainty. The solution

has the form of a modified Gauss-Newton approach that has both speed and accuracy advantages compared to prior published solutions for this particular problem.[108,109]

CSA takes advantage of the fact that the IMLP algorithm was also developed using the cisst libraries, thus making the integration of this module straightforward. Further, the speed and accuracy of the IMLP algorithm help maintain the real-time aspect of the framework.

Coherent Point Drift (CPD)

The CPD algorithm is another probabilistic method, developed by Myronenko et al.,[51] that can be used for both rigid and nonrigid point set registration. For the purpose of CSA, only the nonrigid registration is used. CPD considers the alignment of two point clouds as a probability density estimation problem. The moving point set is assumed to be a Gaussian Mixture Model (GMM) and GMM centroids are fit to the fixed point set by maximizing the likelihood. The GMM centroids are forced to move coherently as a group to preserve the topological structure of the point sets. Coherence constraints are imposed to derive an optimal nonrigid transformation. The details of the coherent constraints are detailed in [51].

3.4.2 Workflow of Model Registration

The communication flow is very simple and straightforward as shown in Figure 3.19. First, the hash grid along with some necessary parameters for registration are initialized. This component is derived from *mtsConntinuousTask* (described in Section 2.3), thus, commands of all the provided interfaces and events of all the required interfaces are dequeued at the start of the run loop. The latest set of surface information from the GP component (X_s and K_s) is fetched and added to the hash grid. Next, the IMLP algorithm is executed on all available sample points stored in the hash grid. The output of the algorithm is stored in the component's "StateTable", which is later fetched by the "Intermediate Interface".

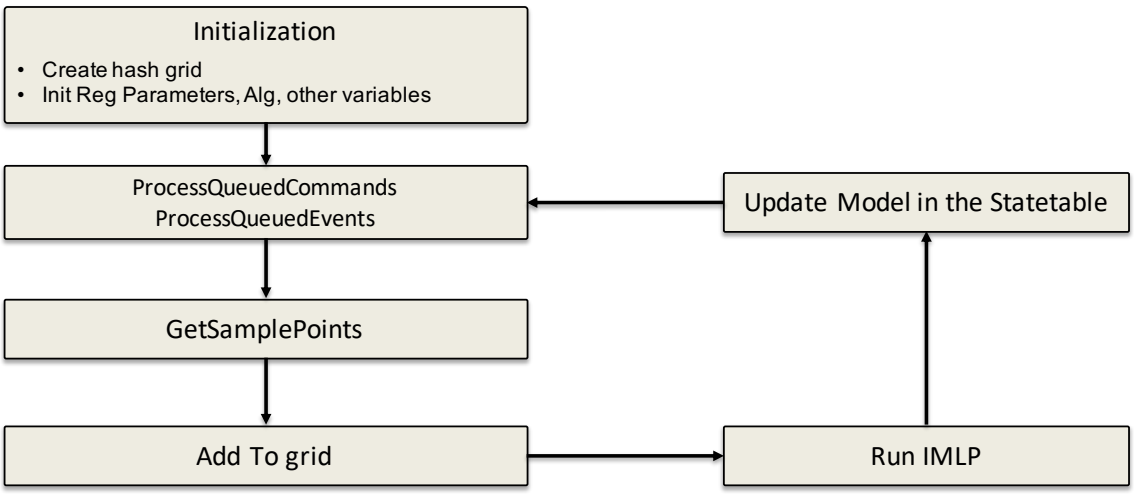


Figure 3.19: Registration Workflow

3.4.3 Data storage and retrieval

As mentioned in the previous section, spatial hash grids are very useful when it comes to data collection and retrieval. Therefore, they have also been used for storing the incoming surface data from the GP component and for fast retrieval for the rigid registration. Output of the GP component is a set of surface points (X_s) and corresponding stiffness values (K_s) that are used in IMLP. This information is stored in hash grids based on spatial locations $x_s \in X_s$ using Equation 3.14. Following is the information stored in each grid location :

- *vct3* **SurfacePos**
- *double* **SurfaceStiffness**

At any time, only a single pair of location and stiffness is allowed to be stored in a grid location. New information always replaces the old information stored in a particular grid location. When the IMLP algorithm is ready to load in the next set of input sample points, the latest information in all the grid locations is retrieved for registration.

3.4.4 Example

Let us assume that some form of preoperative data of a kidney phantom is already acquired. For the purpose of demonstration, a Stereolithography (STL) model representing an undeformed silicone phantom, as shown in Figure 3.20, is used. A point

cloud, M_P, is obtained from this STL model, by means of random sampling, representing the preoperative data. Our goal is to align the point cloud, M_P, onto a set of points, X_s, generated from the GP component during a palpation or an exploration task on an intraoperative model, shown in Figure 3.21. The assumption here is that the surgeon would have access to preoperative data of the kidney (this example), and the intraoperative data is gathered by the means of palpation or exploration. Based on the intraoperative and preoperative data, the *Modeler* updates the model of the kidney for the surgeon to visualize the abnormalities of the tissue.

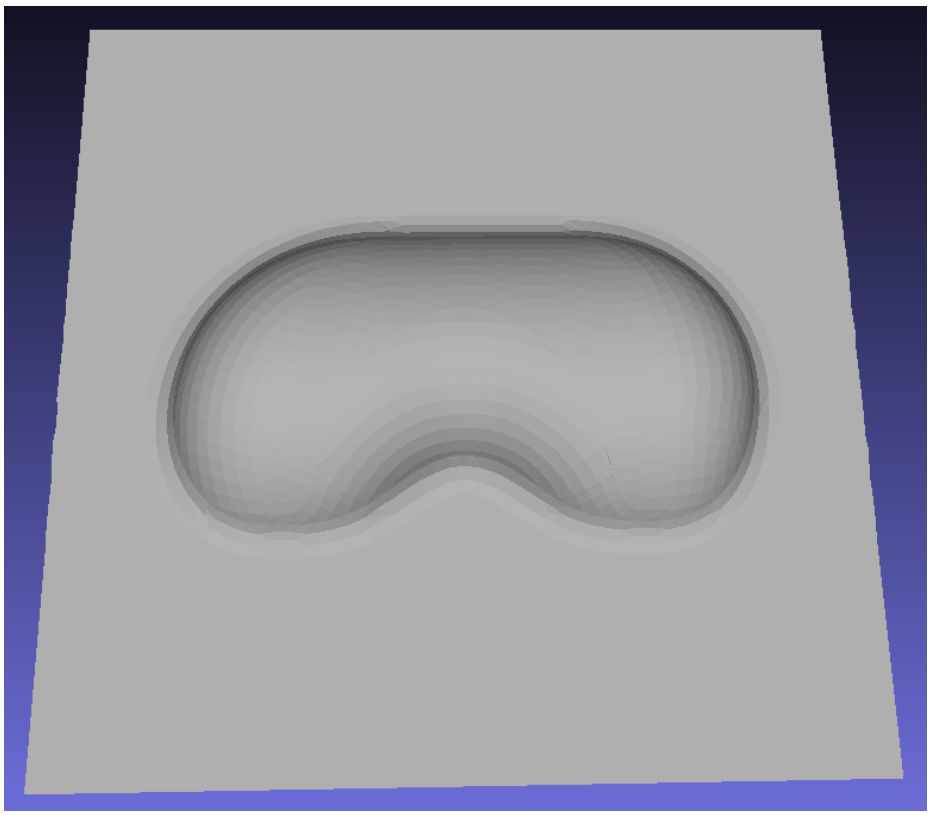


Figure 3.20: STL Model of an undeformed kidney phantom from which point cloud, M_P, is sampled

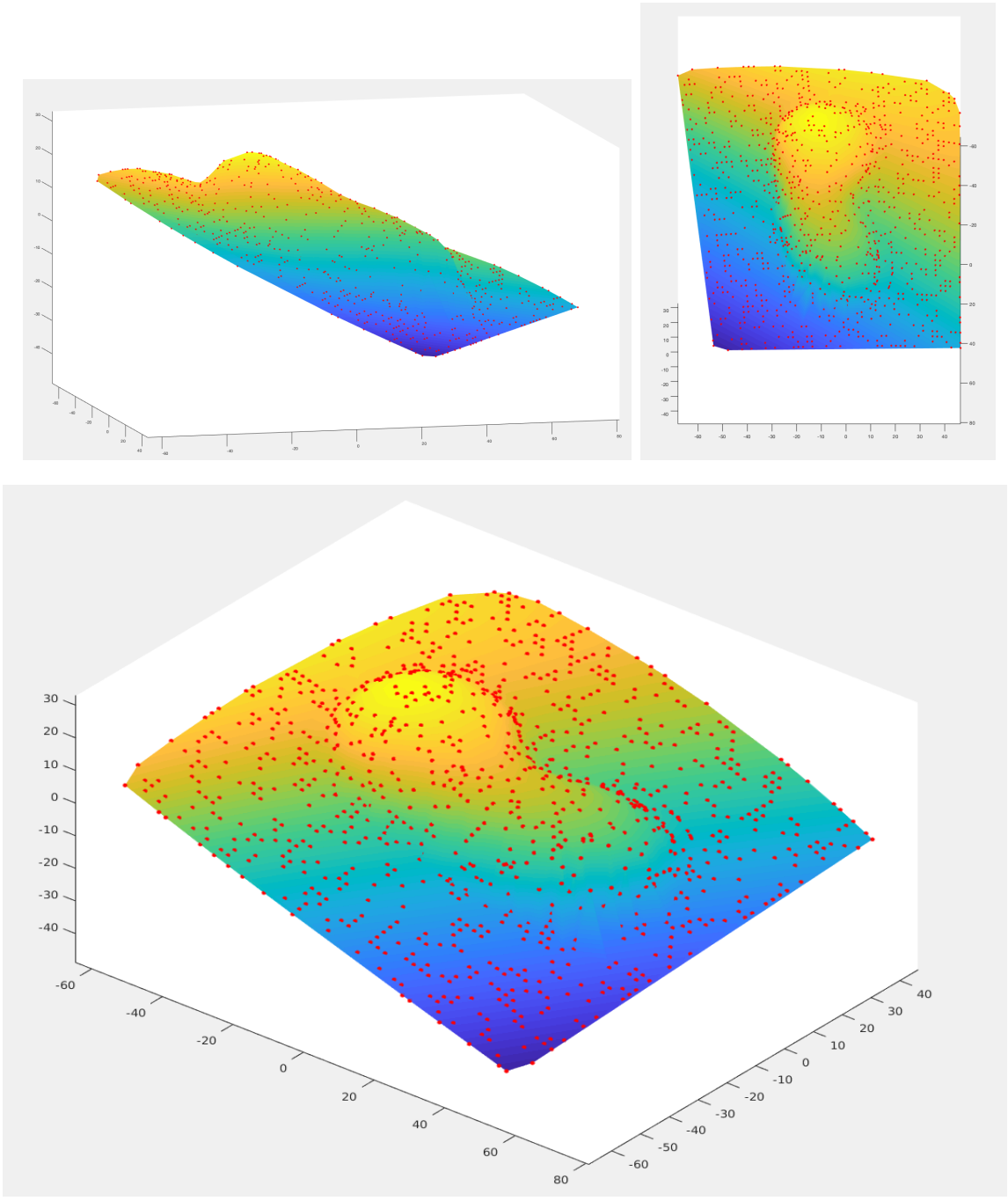


Figure 3.21: Deformed point cloud (X_s) : Red dots represent the points from which a mesh was interpolated for display purposes.

For this purpose, first M_P is registered to X_s using IMLP, resulting in X_s^T as shown in Figure 3.22. X_s^T denotes the set of transformed points of X_s that are aligned with the M_P with minimal total registration error (TRE) based on IMLP.

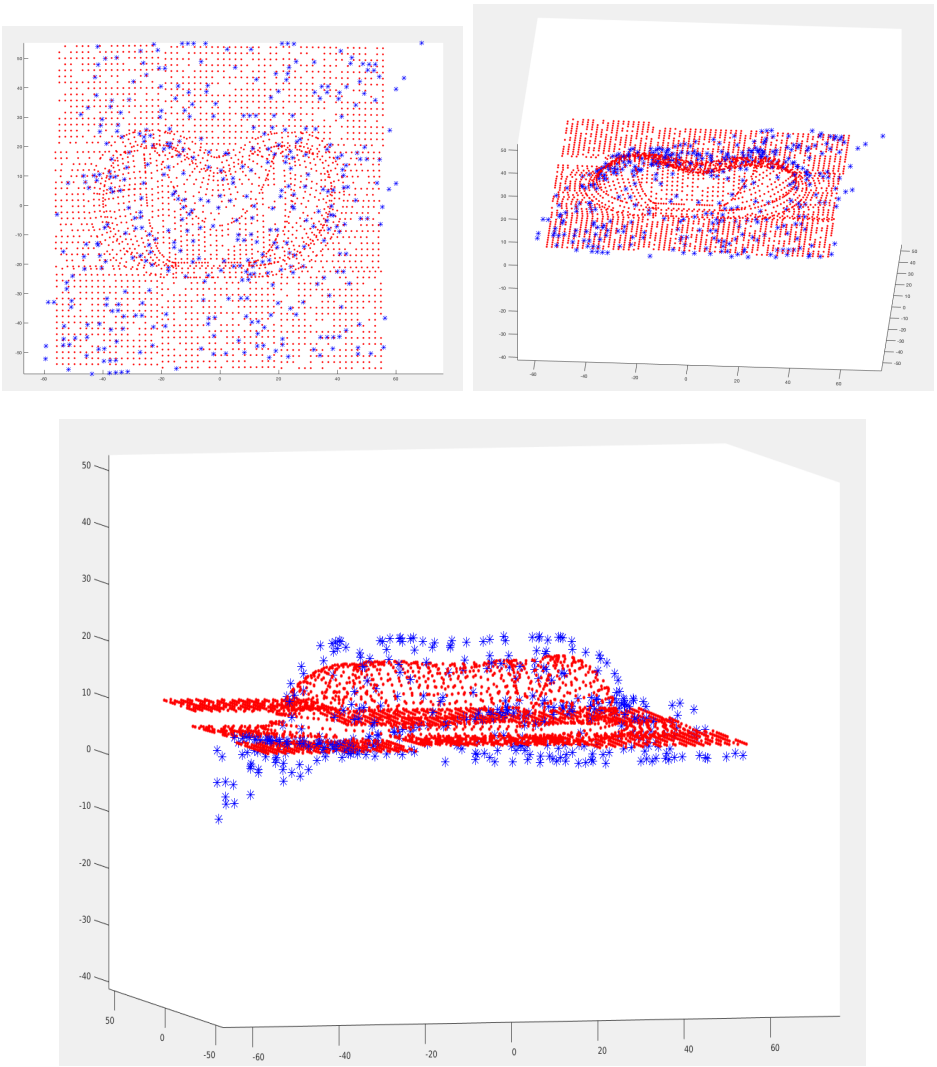


Figure 3.22: Result of IMLP after registering the deformed point cloud to align with the undeformed point set. The red dots represent the preoperative point cloud, M_P, and the blue stars represent the transformed intraoperative point cloud, X_s^T

Now, CPD is used to perform a nonrigid registration to align M_p onto the transformed point cloud X_s, the result of which is shown in Figure 3.23.

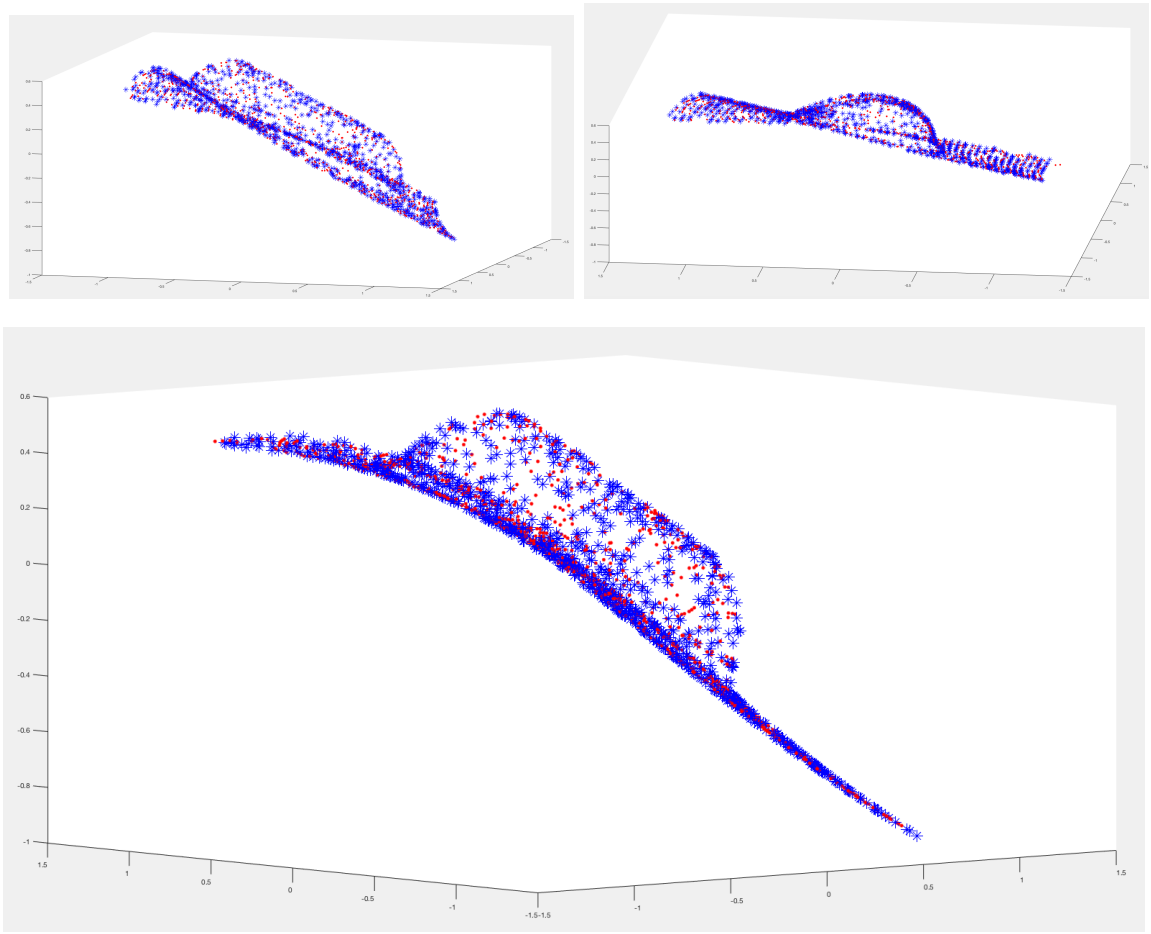


Figure 3.23: Result of CPD after deforming M_p onto X_s.

Figure 3.24 shows the initialization and the final result of the complete registration process.

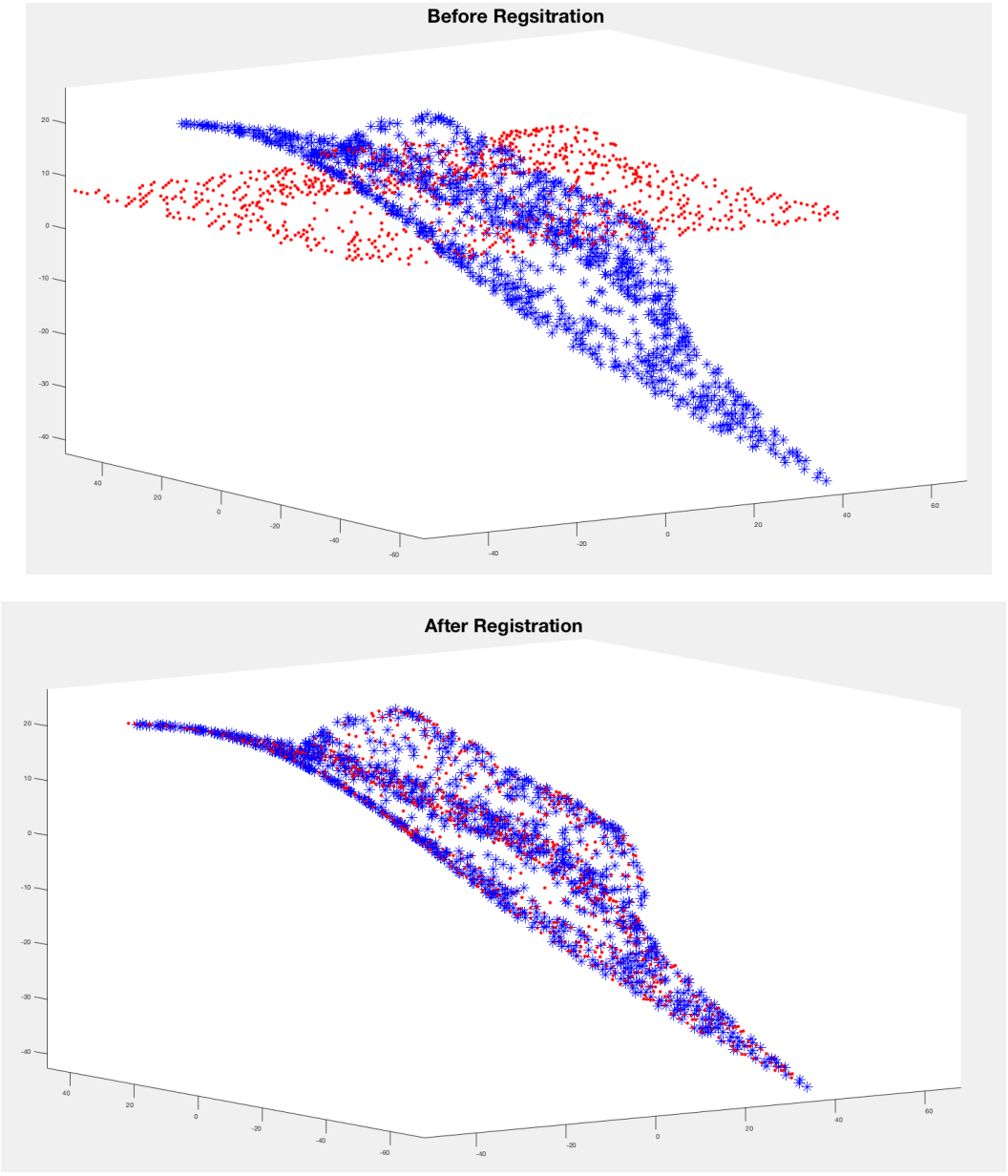
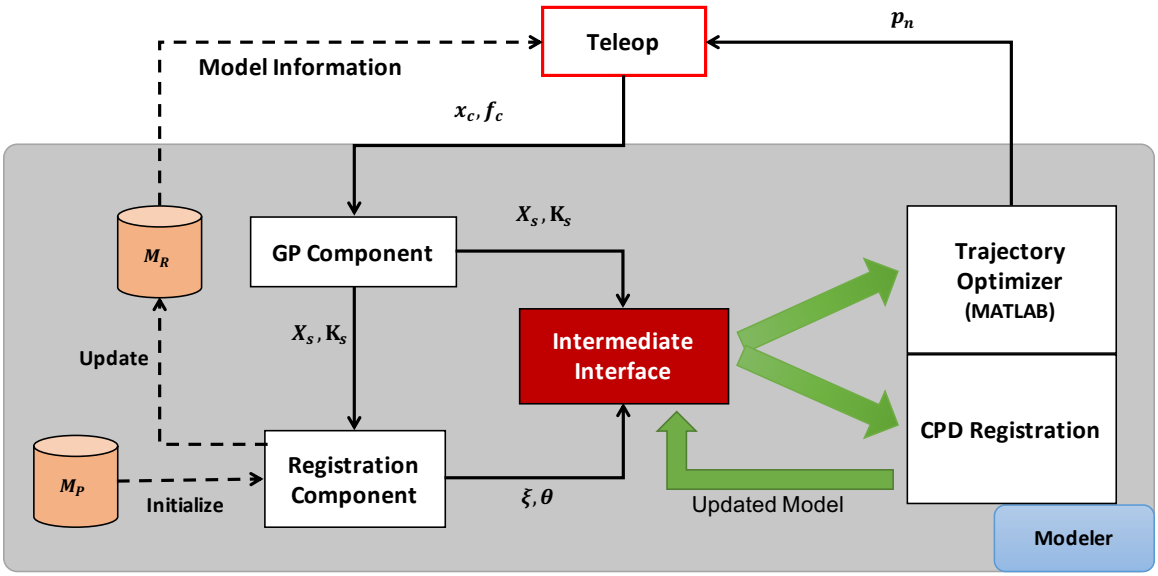


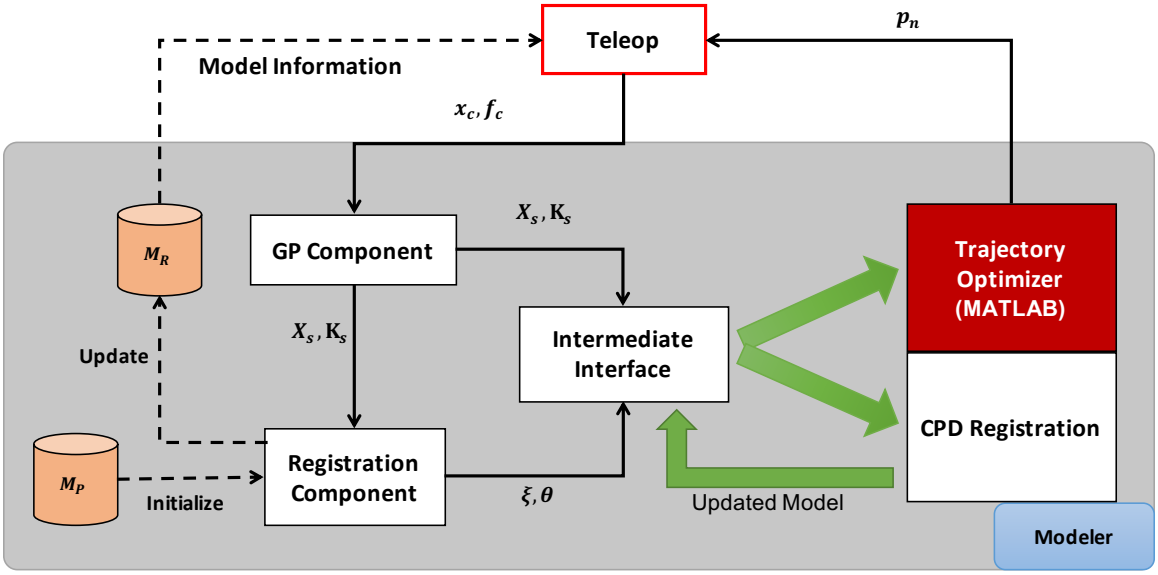
Figure 3.24: Before and after registration result: Red points corresponds the the preoperative data, M_P, and the blue stars corresponds to the intraoperative data, X_s.

3.5 Intermediate Interface



The intermediate interface allows for efficient data transmission from C++ to MATLAB's workspace, and further, allows the C++ program to execute MATLAB scripts directly. This component uses those libraries of MATLAB, C/C++ and Fortran Engine[110] containing routines that allow you to call MATLAB from your own program using MATLAB as a computation engine. The MATLAB Engine operates by running in the background as a separate process from the C++ program. This allows the developers to configure their workstations by having their user interface on one workstation and performing the computations on a faster machine located elsewhere on the network.

3.6 Trajectory Optimization



CSA provides a means of exploring various trajectory optimization techniques to estimate the next optimal direction or trajectory based on user-specified cost functions. Here, some initial work on trajectory optimization using the offline GP approach of estimating organ geometry and stiffness is discussed. The key advantage of using GPs to estimate object properties, including stiffness and geometry, lies in the availability of uncertainty confidence intervals. By predicting both the mean and the variance, one can adaptively guide palpation towards informative regions of interest. This is accomplished through local continuous tool-tip path optimization aiming to 1) reduce the overall model uncertainty, or 2) locate abnormally stiff features as quickly as possible.

3.6.1 Formulation

The optimized adaptive sampling is formulated as a stochastic optimization over trajectories parametrized using a finite dimensional vector ξ. The vector encodes a sequence of n_s arcs in 3-D space and is defined as

$$\xi = (v_1, \omega_1, d_1, \ldots, v_{n_s}, \omega_{n_s}, d_{n_s}),$$

where v_i denotes the horizontal forward velocity, ω_i denotes the horizontal angular velocity, and d_i is the rate of change of depth. The motion is generated by assuming that the tip moves with constant velocity (v_i, ω_i, d_i) during time interval $[t_{i-1}, t_i]$, for each $i = 1, \ldots, n_s$ with $t_0 = 0$, and t_{n_s} provided as a fixed parameter.

Under this parametrization at time $t \in [t_i, t_{i+1}]$, the position of the tip $p(t)$ and its heading angle $\theta(t)$ are

$$p_x(t) = \begin{cases} p_x(t_i) + \frac{v_{i+1}}{\omega_{i+1}}(\sin\theta_{i+\Delta} - \sin\theta_i), & \text{if } \omega_{i+1} \neq 0 \\ p_x(t_i) + v_{i+1}\Delta t_i \cos\theta_i, & \text{otherwise} \end{cases}$$

$$p_y(t) = \begin{cases} p_y(t_i) + \frac{v_{i+1}}{\omega_{i+1}}(\cos\theta_i - \cos\theta_{i+\Delta}), & \text{if } \omega_{i+1} \neq 0 \\ p_y(t_i) + v_{i+1}\Delta t_i \cos\theta_i, & \text{otherwise} \end{cases}$$

$$p_z(t) = p_z(t_i) + \Delta t_i d_{i+1}$$

$$\theta(t) = \theta(t_i) + \Delta t_i \omega_{i+1},$$

where $\Delta t_i = t - t_i$, $\theta_i = \theta(t_i)$, and $\theta_{i+\Delta} = \theta_i + \Delta t_i \omega_{i+1}$.

Two different cost functions were defined for given candidate paths $p(t)$: one minimizing the variance of stiffness, and thus, improving the overall model quality; and the other seeking maximum stiffness. The costs are defined, respectively, by

$$J(\xi) = \begin{cases} \int_{t_0}^{t_f} -V_\kappa(p(t))dt, & : \text{variance}, \\ \int_{t_0}^{t_f} -\mu_\kappa(p(t)) - \beta\sqrt{V_\kappa(p(t))}dt & : \text{UCB}, \end{cases}$$

where β is a user-defined constant, determining the upper confidence bound (UCB) on the stiffness estimate, and is used to balance between visiting known stiff regions and exploring uncertain, but potentially stiffer, regions. $\beta = 1.96$ was set, corresponding to 95% confidence in the stiffness estimate. Both costs are computed by approximating the integrals using a finite number of points p_i along $p(t)$, and predicting the mean $\mu_\kappa(p_i)$ and variance $V_\kappa(p_i)$ from the current GP models.

The actual optimization of $J(\xi)$ is performed using the cross-entropy method,[111,112] which is a global optimization method that places a Gaussian density over ξ and iteratively updates this density until it has concentrated around the optimal ξ^* minimizing the cost function. This method was chosen since it can handle non-smooth and highly irregular cost landscapes such as those that occur during non-parametric estimation.

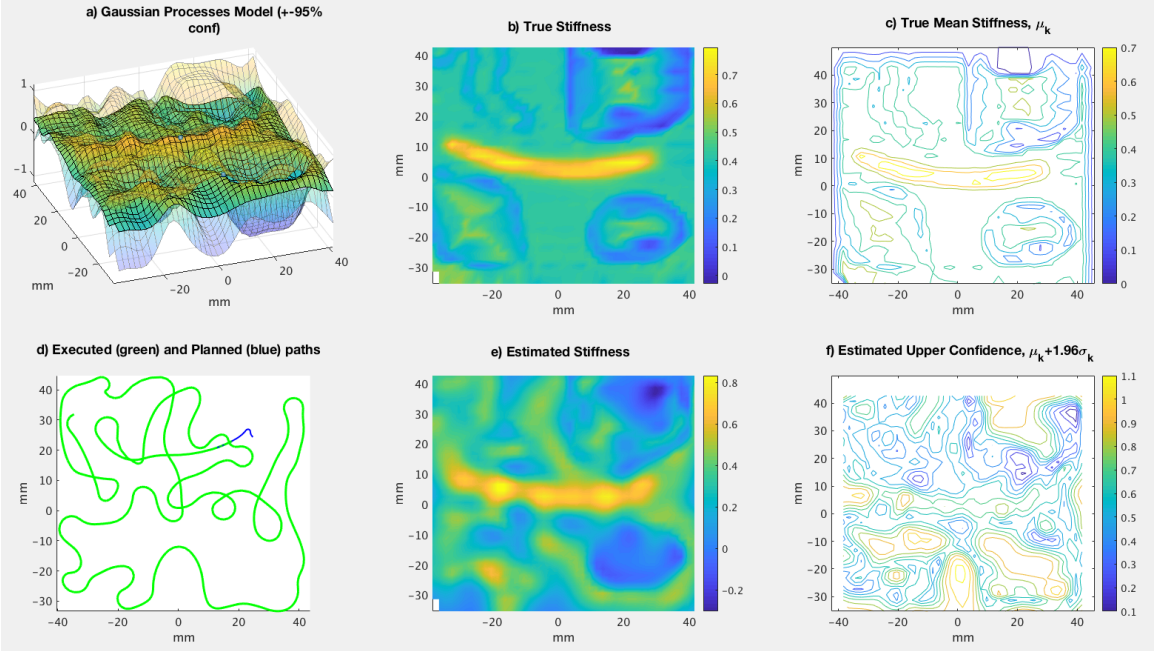


Figure 3.25: Adaptive sampling: a) GP Model over stiffness; b) True stiffness; c) Contour plot of the true stiffness; d) Executed trajectory path; e) Estimated stiffness f) Estimated upper confidence level of stiffness

3.6.2 Experiment

Adaptive sampling was tested by initializing at a random location. It was interesting to observe, in the executed path shown in Fig 3.25(d), that the simulated trajectory constantly follows a spiral motion at stiff regions or unexplored regions. This is identical to what a person would intuitively do to characterize the surface properties. This shows that the trajectory, simulating the motion of the probe, adaptively changes the trajectory parameters to minimize the uncertainty at unexplored and stiff regions. Fig 3.25e reveals the underlying stiff feature based on the executed trajectory in Fig 3.25d. However, a significant noise is observed in the estimated stiffness field, and it is likely due to incorrect force interpolation at the current location $p(t)$.

This work was not further explored as other members of the project took over the trajectory optimization module and their work can be found in [70, 113–115].

3.7 Contributions

The contributions reported in this chapter include:

1. A novel method for simultaneously estimating organ geometry and tissue stiffness using Gaussian Processes, that is independent of the palpation strategy (discrete or continuous).

2. Online estimation of organ geometry and tissue stiffness using spatial hash grids

and local Gaussian Processes, during continuous palpation.

3. Near-video frame rate update of the stiffness map for display purposes based on online estimation, the results of which were evaluated and also compared with the offline method.

3.8 Published Work

Material from this chapter has appeared in the following publication:

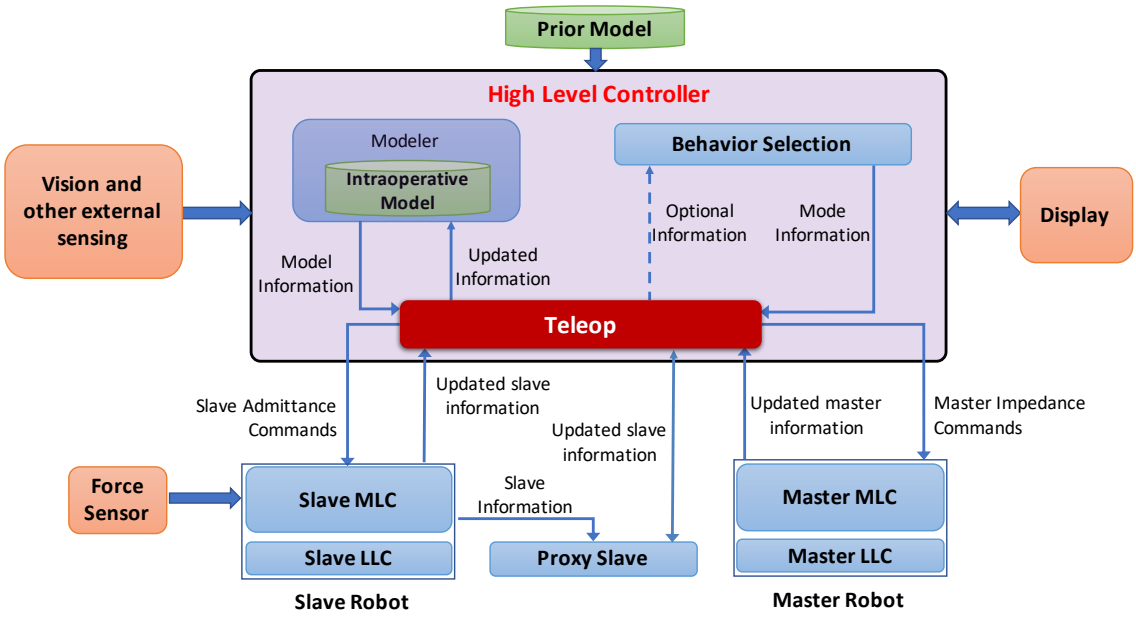
1. P Chalasani, A Deguet, P Kazanzides, RH Taylor, "A Computational Framework for Complementary Situational Awareness (CSA) in Surgical Assistant Robots," in 2018 Second IEEE International Conference on Robotic Computing (IRC), 9-16

2. P. Chalasani, L. Wang, R. Yasin, N. Simaan, and R. H. Taylor, "Preliminary evaluation of an online estimation method for organ geometry and tissue stiffness," in 2016 IEEE Robotics and Automation Letters, vol. 3, no. 3, pp. 18161823.

3. P. Chalasani, L. Wang, R. Roy, N. Simaan, R. H. Taylor, and M. Kobilarov, "Concurrent nonparametric estimation of organ geometry and tissue stiffness using continuous adaptive palpation," in 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 4164-4171

Chapter 4

High-Level Controllers - *Teleop*



Teleop is part of the high-level controller which maintains the state information of various components and is responsible for managing communications between those components. As mentioned earlier, the whole CSA paradigm is based on Model

Mediated Teleoperation (MMT). MMT has been initially proposed to address the stability and transparency issues during communication delays.[48,116,117] In the MMT approach the user telemanipulates on a local object model employed on the master side. This model is monitored and updated to approximate the slave environment. Haptic feedback is computed on the basis of the local model on the master side. Many groups have developed methods for accurate estimation of the remote environment to provide stable and transparent teleoperation.[48,118,119]

Research on MMT started since the late 1980s, and at that time the predictive display approach was developed to visually compensate for the communication delay.[120,121] The first prototype of MMT was proposed by Hannaford,[116] which is very similar to the state-of-the-art MMT approach. Later, the use of predefined models in MMT systems shows some level of improved stability against communication delay, and also of robustness against small modeling errors.[122] However, using a predefined model potentially results in modeling error. Alfi *et al.*[123] proposed a modified structure to improve system robustness against uncertainties of time delay and model parameters.

Mitra and Niemeyer[48] were the first to use the name MMT to indicate an alternative type of information exchange between the master and slave. The user holds the master which is connected to a proxy slave. The proxy slave acts as a stand-in for the real slave robot. The user interacts locally with the haptically rendered model and the master-side forces are computed based on error distance between the proxy slave

and the master. The slave executes both position and force commands based on its contact with the environment and the contact state of the master with the virtual environment. A more comprehensive survey on MMT can be found in [124].

4.1 Model mediated teleoperation (MMT) in CSA

The idea of MMT approach, employed in the CSA, is inspired by the original work done by Mitra and Niemeyer.[48] Many researchers have developed techniques to improve the model estimation, to increase the stability during communication delay, however, the underlying information exchange between the master and slave remained the same. The proposed MMT paradigm in this dissertation closely resembles the concept of using a haptic proxy to interact with the master during MMT, however, we provide a different way of correcting the model-reality mismatch, apart from updating the environment itself.

In [48], Mitra and Niemeyer control a proxy slave using the master robot. By construction, the model of the environment on the master side will track the environment on the slave side. The master sends force and motion commands tracked by the slave robot, and the slave robot updates its model of the environment by monitoring the contact with the floor. The master model then generates a haptic virtual floor based on contact information sent by the slave. Since the proxy slave embodies the

slave mechanism, it would track the master motion when in free space and will stop at the model boundary when in contact. Further, the model of the virtual floor is updated based on various transition states, as described in [48]. The slave robot also switched between position and force control based on the contact information from the master. This complete workflow allows the user to have smooth haptic feedback even with communication delay. The authors explained this mismatch/correction criteria using a single DOF case study. However, we are dealing with 3D models and have no communication delays in surgical teleoperation. This allows us to modify the mismatch/correction criteria for smooth and faster haptic updates and cover the cases when the physical slave is palpating into the organ surface.

In our approach, we do not have a dynamic proxy slave, rather a simple position controlled proxy. The proxy slave always follows the master motion and is allowed to penetrate the virtual environment. The master force or the compliance force is computed based on the interactions of the proxy slave with the virtual environment. The slave robot is controlled based on hybrid force-motion control, such that the commanded motion from the force controller is projected in the normal direction of the force vector. Similarly, the commanded motion from the position controller is projected in the tangential direction of the force vector.

Figure 4.1 shows the communication flow in MMT. Since the user is teleoperating the proxy slave, the virtual environment will eventually drift from reality when there is an interaction with the task model. This is due to the difference in control of

the slave and proxy slave robots, as described in Sections 5.1 and 5.2, respectively. The various mismatch states, along with their correction techniques, are described in Section 4.1.1.

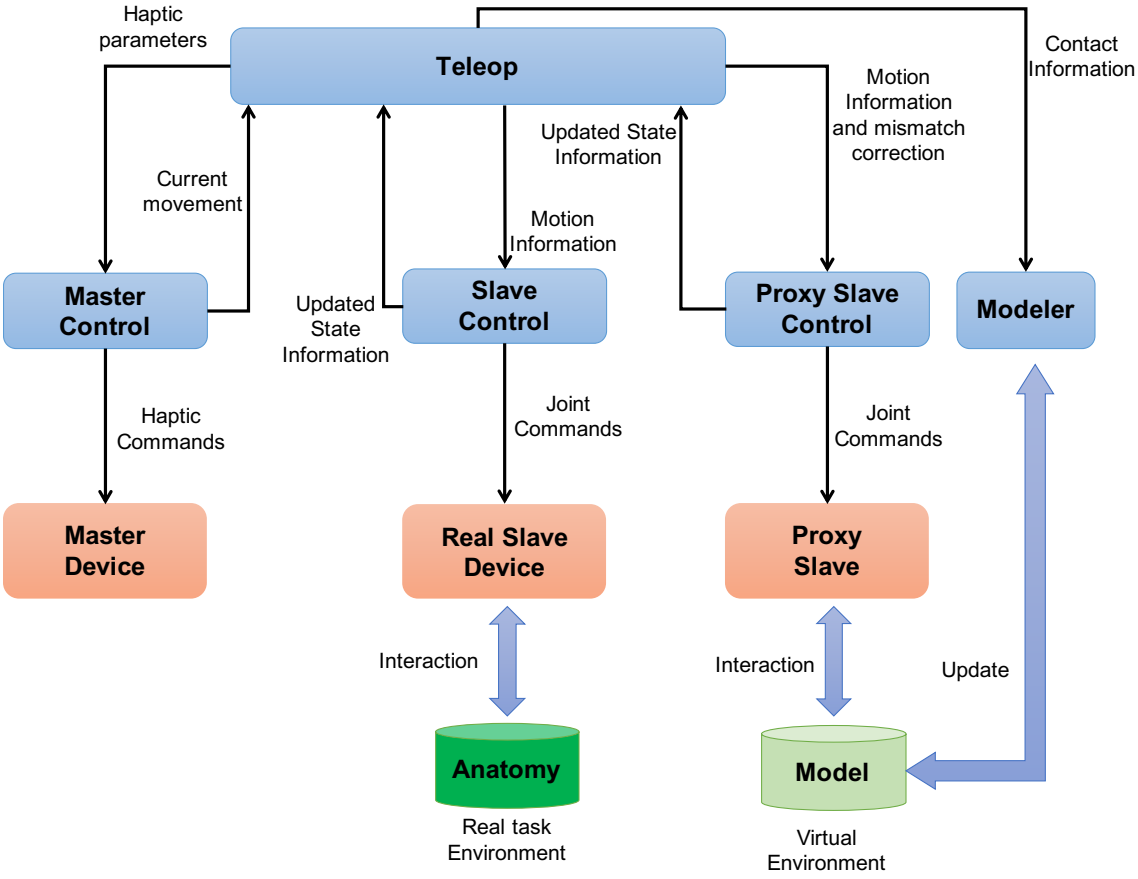


Figure 4.1: High level communication flow of MMT in CSA

4.1.1 Mismatch Scenario

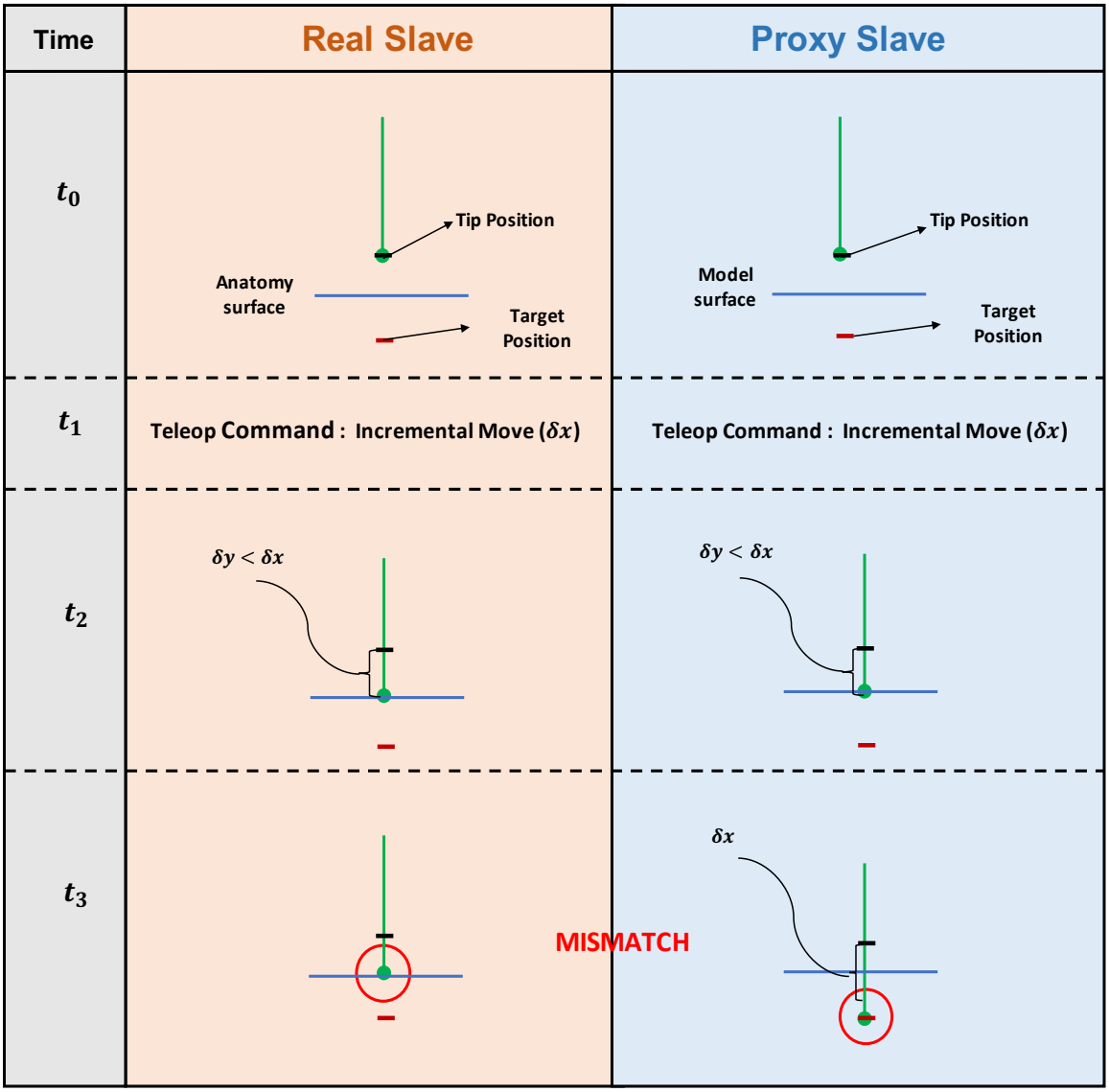


Figure 4.2: A sample mismatch scenario between real slave's tip position and proxy slave's tip position

- At time step t_0 the slave robot and the proxy slave robot are at the exact same location in the real and the virtual environment, respectively. Let us assume a

perfect registration of the anatomy with the real salve and model of the anatomy with the proxy slave. At t_0, tip positions of both the slaves are δy units away from the surface of their respective anatomies. (blue line)

- Based on master movement, at time step t_1, both receive an incremental move, $\delta x > \delta y$, command from the *Teleop*. Both the slave and proxy slave components will feed in the commanded motion into their respective control algorithms.

- At time step t_2, slave robot makes contact with the surface of the anatomy (real world) after moving δy units. Similarly, the proxy slave also makes contact with the surface of the model (virtual world) after moving δy units. Moving forward, the real slave, equipped with force sensing modality, will send force information to the *Teleop* based on contact with the anatomy. *Teleop* will then generate new sets of motion commands for the real slave to constrain the motion from damaging the anatomy. Since the proxy slave has no such force sensing modality, it will send distance information, ϵ, from the tip position to the surface.

- At t_3, slave robot servos at the point of contact maintaining minimal contact force with the surface of the anatomy. However, the proxy slave will complete the rest of the trajectory based on the motion command sent by the *Teleop*.

- The slave robot reaches the surface of the model and servos at the point of contact maintaining minimal contact force. However, proxy slave robot, obeying position control, will cross the surface in the virtual environment, creating a

mismatch between the joint state of the slave robot and the proxy slave.

This creates a mismatch between the joint states of the real slave and the proxy slave, even though the model is accurate.

4.1.2 Mismatch Correction

To correct the mismatch between the joint states of the real and proxy slave, the *Teleop* is operated in four operational modes. These modes are not selected by the user, rather are switched based on contact states of the real slave and the proxy slave robots. Based on the contact information, necessary admittance gains are sent to the *Slave MLC*, as described in Section 5.1. Similarly, compliance forces are also sent to the *Master MLC*, as described in Section 5.3.

However, the user can modify the interaction behavior based on task specifications.

1. `PROXY_CONTACT`: This is the state of *Teleop* when the proxy slave is in contact with the situational model but the slave robot is not in contact with the anatomy.

   Correction: Compliance forces are sent to the *Master MLC* for haptic rendering. Desired force (F_d) is calculated and sent to the *Slave MLC* according to the following equation:

$$F_d = k_d \vec{\epsilon}_s \tag{4.1}$$

where $\vec{\epsilon_s}$ is a vector defining the depth of the proxy slave's tip inside the surface and k_d is a scaling factor resembling stiffness of the tissue. This correction will eventually change the teleop state to `FULL_CONTACT`, when pushing into the model, or `NO_CONTACT`, when transitioning into free space.

2. `SLAVE_CONTACT`: This is the state of *Teleop* when the proxy slave is not in contact with the situational model but the slave robot is in contact with the anatomy. *Correction*: Send zero or minimal desired force to the *Slave MLC*, along with necessary admittance gains. The *Teleop* state will switch to `FULL_CONTACT`, if the user moves towards the model or `NO_CONTACT`, if the user moves away from the anatomy.

3. `FULL_CONTACT`: This is the state of *Teleop* when the proxy slave is in contact with the situational model and the slave robot is also in contact with the anatomy. *Correction*: Compliance forces are sent to the *Master MLC* and admittance gains are sent to the *Slave MLC*. Additionally, the direction of the desired force is calculated based on the direction of the sensed contact force. This will make sure that the force reference direction is close to the surface normal direction.

4. `NO_CONTACT`: This the state of *Teleop* when both the slave robots are in free space. Both the slaves follow the master motion and no correction is required in this state.

This design allows the system to dynamically detect and correct the mismatch be-

tween the virtual environment and reality. Simultaneously, the system also provides smooth haptic rendering when interacting with the situational model (described in Section 5.3).

4.2 Component Connections

Teleop is implemented as a periodic task (*mtsTaskPeriodic*) with an update rate of the loop set to ∼500 Hz. *mtsTaskPeriodic* is derived from *mtsTask* and all tasks derived from *mtsTask* have a *Run* method which is called in a loop based on the provided periodicity. Further, it has the capability of adding provided and required interfaces to an existing component. CSA relies on the fast data transfer using interface communication, via provided and required interfaces. Figure 4.1 demonstrates the interface connections between the *Teleop* component and all other components in CSA. To retrieve information from other components, *Teleop* has to add a required interface to its component and connect that interface to the provided interface of the component that has the necessary information. In this case, required interface of the *Teleop* is connected to the provided interface of the *Master MLC*, *Slave MLC* and *Proxy Slave* components to retrieve contact information, state information and access few interface functions.Similarly, *Teleop* also has a provided interface that provides access to model information, state information of all the arms and other useful debugging information. *Modeler* component connects to this provided interface to retrieve

force and position information of the slave. Further, *ROS Bridge* also connects to the *Teleop's* provided interface to broadcast useful information to applications running outside the cisst-saw environment.

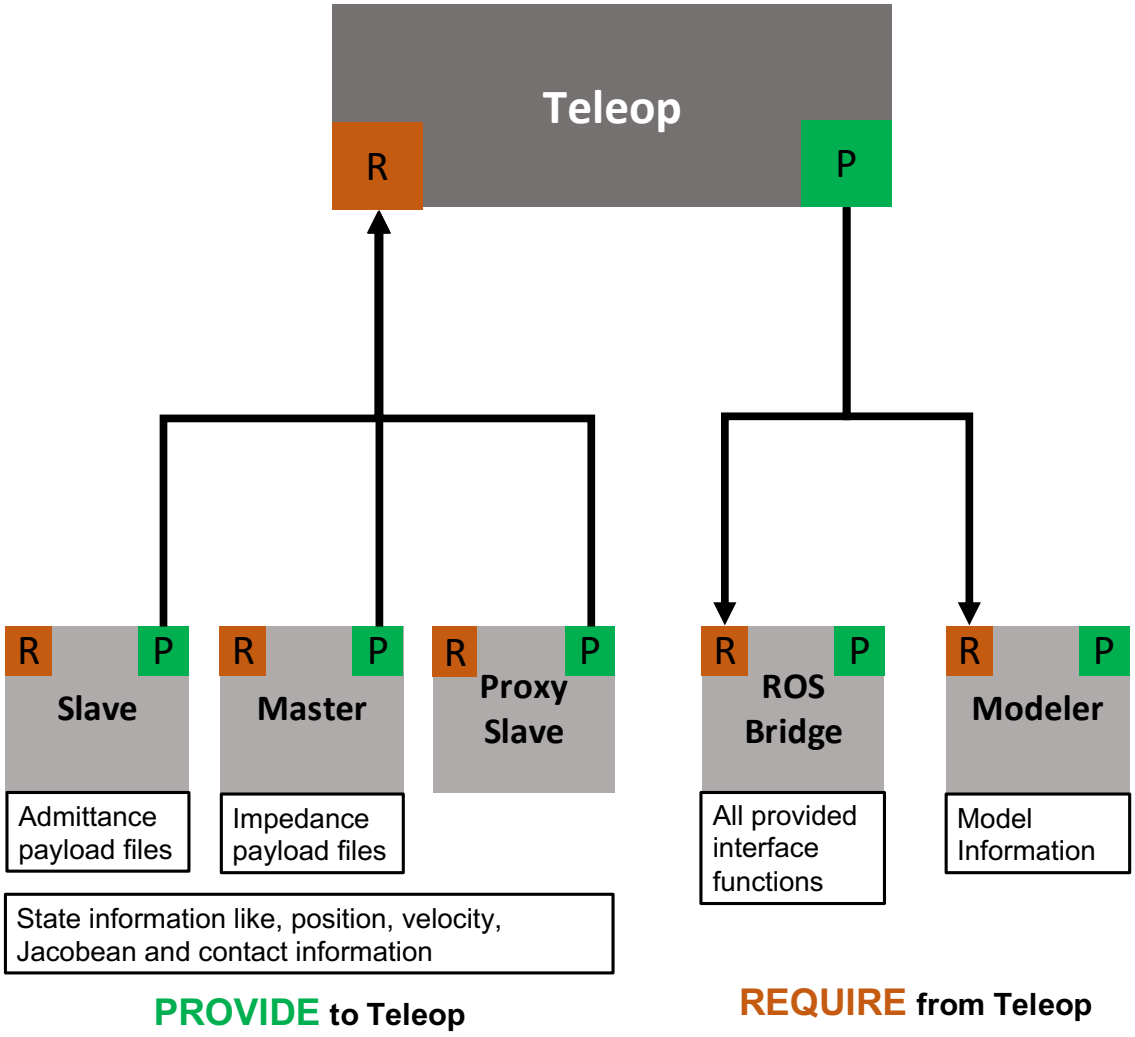


Figure 4.3: *Teleop* interface connections

4.3 Contributions

Principal contribution reported in this chapter include:

1. Application of model mediated paradigm and provided ways to correct the model-reality mismatch.

4.4 Published Work

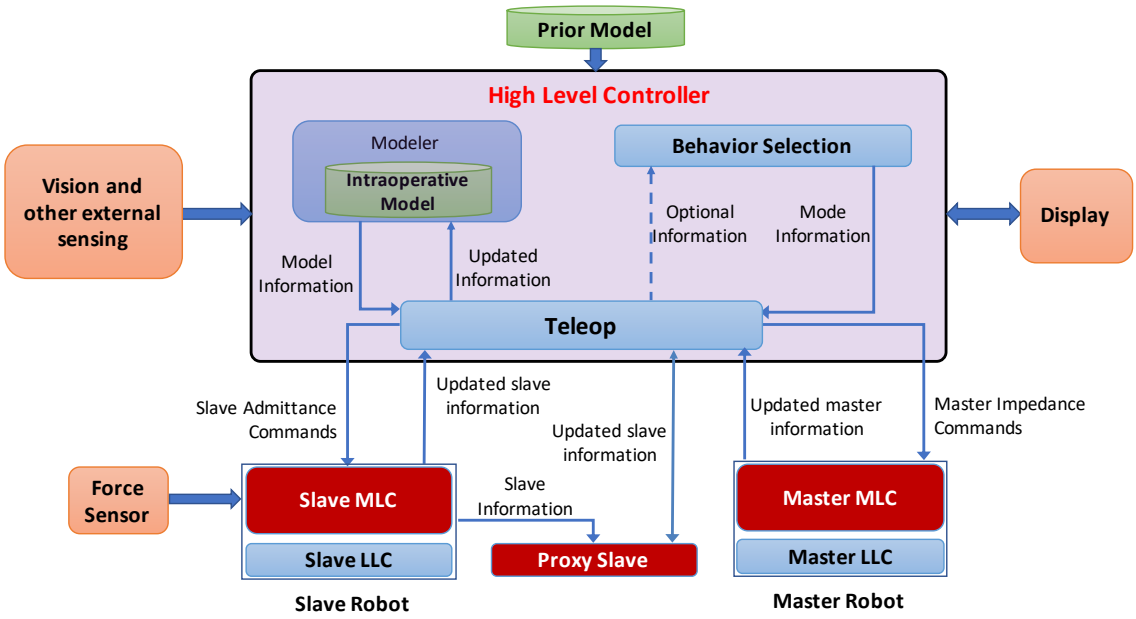
Material from this chapter has appeared in the following publication:

1. P Chalasani, A Deguet, P Kazanzides, RH Taylor, "A Computational Framework for Complementary Situational Awareness (CSA) in Surgical Assistant Robots," in 2018 Second IEEE International Conference on Robotic Computing (IRC), 9-16

2. L. Wang, Z. Chen, P. Chalasani, R. M. Yasin, P. Kazanzides, R. H. Taylor, and N. Simaan, "Force-controlled exploration for updating virtual fixture geometry in model-mediated telemanipulation," in 2017 Journal of Mechanisms and Robotics, vol. 9, no. 2, p. 021010

Chapter 5

Mid-Level Controllers



The da Vinci Research Kit (dVRK) is one of the widely used research platforms for teleoperation. The mid-level controller of the CSA is implemented in such a way that it extends the functionality that of the dVRK mid level control by adding more

features and capabilities to help complete the MMT paradigm. This chapter presents the three MLCs of CSA, namely *Slave*, *Proxy Slave* and *Master*, that are responsible for the control of their respective robots/arms.

5.1 CSA Slave

The *Slave MLC* is responsible for control of the slave manipulator. This component communicates with the *Teleop* component and the *Slave LLC*, which communicates with the slave manipulator. The *Slave MLC* runs as a clock-driven process at a repetition rate of approximately 100-500 Hz. The *Slave LLC* typically may run at a faster duty cycle (such as 2-3kHz). Almost all of the hardware dependencies will be managed in the LLC, although there may need to be some changes in the MLC to accommodate for special hardware needs.

The *Slave MLC* receives admittance and virtual fixture specification commands from the *Teleop* and translates them into cartesian or joint position/velocity commands that are passed onto the *Slave LLC*. In some embodiments, the MLC passes on specialized force admittance commands to the LLC, although this function would normally be performed in the MLC. The *Slave MLC* transforms admittance and virtual specifications into a constrained quadratic optimization problem, which also may comprise manipulator-specific constraints such as joint position, velocity, and acceleration limits.

5.1.1 Design Motivation

The *Slave* component of the CSA extends that of the *dVRK* by adding a force controller to the already existing position controller. Further, afew motion primitives are added in the *CSA Slave* to help in palpation tasks. For comparison, Figure 5.1 shows various subroutines in a standard *dVRK Slave* and Figure 5.2 shows the subroutines in a *CSA Slave*. *dVRK Slave* is implemented as a position controller with the slave robot following an absolute position control law. In contrast, *CSA Slave* is implemented as an admittance type controller with the slave robot following a hybrid force-motion (HFM) control law.[125] Therefore, a force sensing component is connected to the *CSA Slave* to provide interaction forces with the environment.

In the case of the *dVRK Slave*, *Teleop* commands the slave to move to a desired position x_d. Based on the current position x_c and the desired position x_d, the control law computes the necessary joint motion q_d to achieve the requested motion. q_d is then sent to the *Slave LLC* for robot movement.

In the case of *CSA Slave*, the component retrieves sensed forces F_c from the force sensing. The *Teleop* sends admittance gain parameters along with the commanded position x_d and desired/limiting force F_d. F_d ensures the robot to servo at a particular force value. x_d and F_d are passed through a palpation block, which adds user-selected motion primitives to the desired force or position. The output of this block, x_d^* and F_d^*, along with x_c and F_c, is fed to the HFM control law, which calculates the necessary joint motion required to satisfy the commanded position and force.

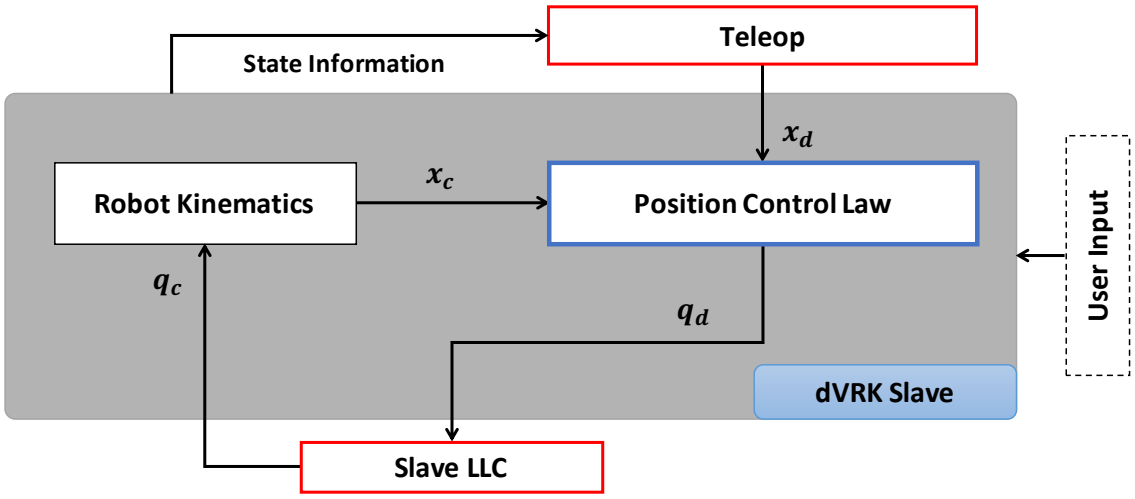


Figure 5.1: Standard dVRK Slave

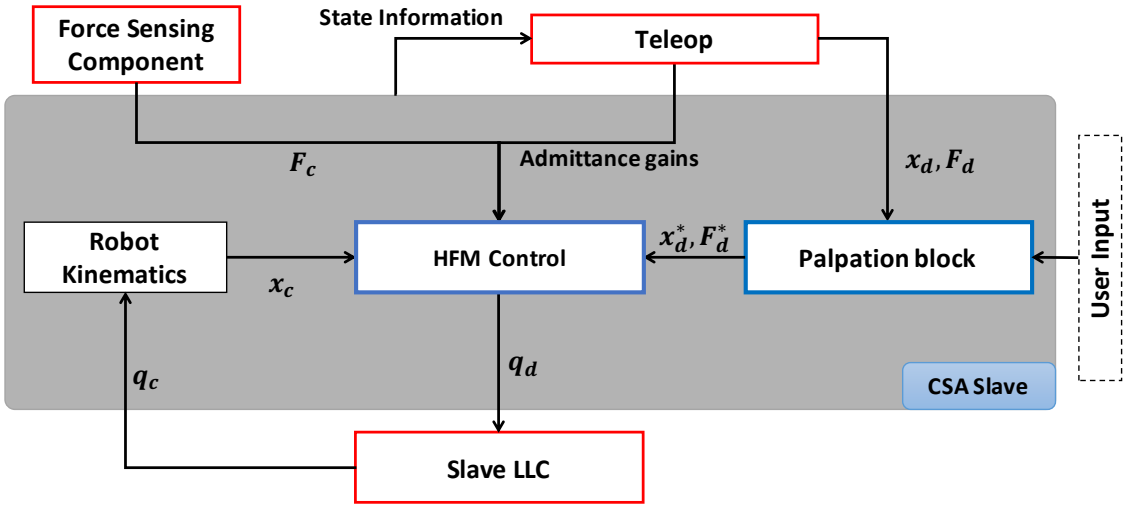


Figure 5.2: Improved CSA Slave

5.1.2 Control Law

As stated earlier, *CSA Slave* is implemented as an admittance controller with the robot following a hybrid force-motion (HFM) control law. This includes both position and force controllers.

Position Controller

The position controller is set up as an objective function to minimize the magnitude of error between the current position x_c and desired position x_d, given as

$$\min_{\delta q} \| J\delta q - (x_d - x_c) \| \tag{5.1}$$

such that,

$$v_l \leq \frac{\delta q}{\delta t} \leq v_u$$

where, J represents the body jacobian of the slave robot and δq corresponds to incremental joint motion. In addition, v_l and v_u correspond to lower and upper joint velocity limits, respectively.

Force/admittance Controller

In MIS procedures, manipulation is done on a very delicate tissue. For this reason, a force controller is an excellent control structure for creating stiff virtual walls prevents the tool from penetrating the tissue. This control can also be set up as an

objective function as follows,

$$\min_{\delta q} \| J\delta q - K_g(F_c - F_d)\delta t \| \tag{5.2}$$

such that,

$$v_l \leq \frac{\delta q}{\delta t} \leq v_u$$

where, F_c is the contact force detected by the force sensor, F_d is the desired/limiting force, δt is the sampling rate of the component in seconds, and K_g is the admittance gain matrix.

Hybrid Force Motion (HFM)

HFM is the combination of the aforementioned position and force controller, with a motion from each controller decomposed in orthogonal directions. The combined objective function would be

$$\min_{\delta q} \| J\delta q - [K_a \underbrace{K_g(F_c - F_d)\delta t}_{\substack{\text{Force} \\ \text{controller}}} + K_p \underbrace{(x_d - x_c)}_{\substack{\text{Position} \\ \text{controller}}}] \| \tag{5.3}$$

$$v_l \leq \frac{\delta q}{\delta t} \leq v_u$$

where, K_a projects the motion from the force controller in the direction normal to the surface, and K_p projects motion from the position controller in the direction perpen-

dicular to the surface normal. The projection matrices are calculated as follows:[126, 127]

$$K_a = N(N^T N)^{-1} N^T$$

$$K_p = I - K_a$$

where, N corresponds to the desired force control direction. The resultant incremental joint motion from Equation 6.3 is then sent to the *Slave LLC* to execute the motion. x_d and F_d are updated if additional motion primitives, as explained in Section 5.1.3, are added to the desired position or force.

5.1.3 Motion Primitives

CSA Slave has a provision for choosing different motion primitives for continuous palpation. Currently, two motion primitives are supported :

(a) **Sinusoidal force reference:** This motion is achieved by having the desired force follow a sinusoidal profile, in the direction normal to the surface.

$$F_d^* = F_d + A * Sin(2\pi\omega t) \tag{5.4}$$

where A, ω, t are parameters of the sine motion. Figure 5.3 depicts the motion of the probe achieved with this motion primitive.

(b) **Sinusoidal motion reference:** This motion is achieved by applying a sinu-

soidal motion reference in the direction of the sensed force vector.

$$x_d^* = x_d + \hat{n}A * Sin(2\pi\omega t) \tag{5.5}$$

where A, ω, t are parameters of the sine motion and $\hat{n} = \frac{F_c}{||F_c||}$ is the direction of the sensed force vector. Figure 5.4 depicts the motion of the probe achieved with this motion primitive.

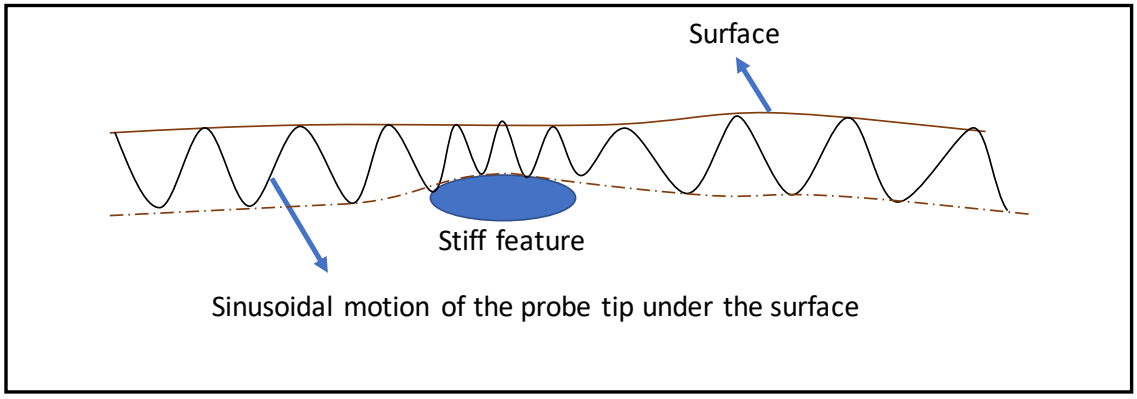


Figure 5.3: Reference force following a sinusoidal profile

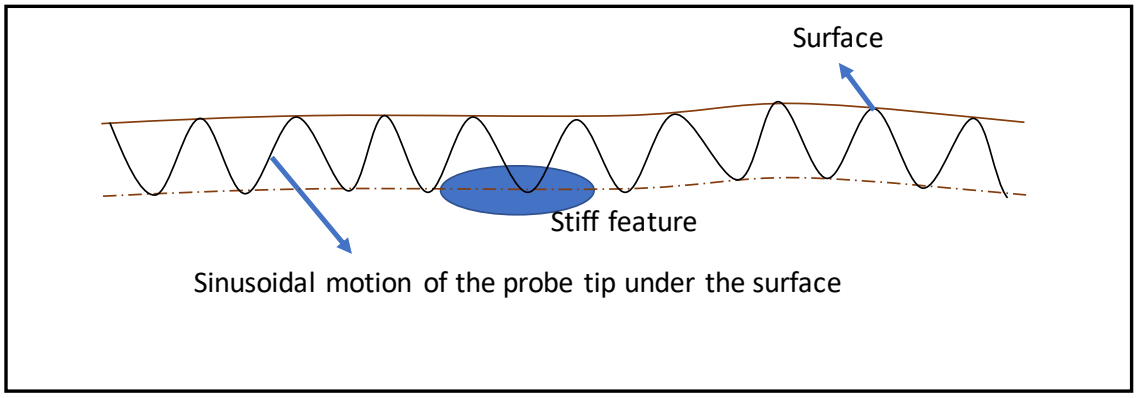


Figure 5.4: Position following a sinusoidal profile

5.1.4 Interface Connections

Figure 5.5 shows the interface connections for the *CSA Slave*. The *PID* and *IO* interfaces provide joint- and encoder-level information, respectively. Further, the *CSA Slave* is connected to the *Force sensing* component that provides contact information for the HFM controller. *Proxy Slave* also connects to the provided interface of *CSA Slave* for intermittent joint synchronization, while homing or in idle state.

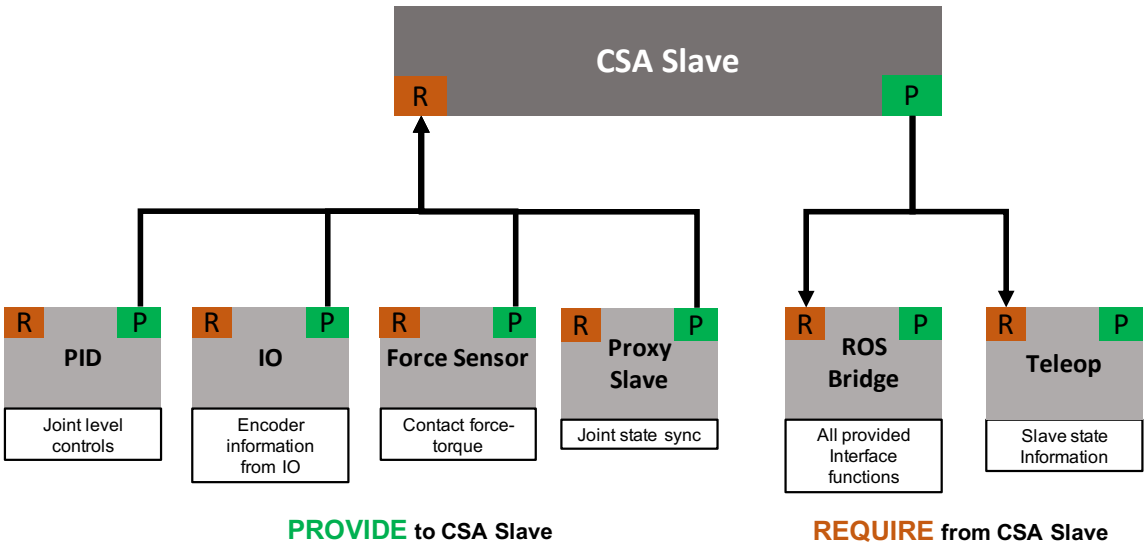


Figure 5.5: *CSA Slave* interface connections

5.2 Proxy Slave

As stated before, CSA focuses on interacting with the situational model and uses the model interactions along with the sensory information to update the model of the task environment. The situational model in the virtual environment is registered

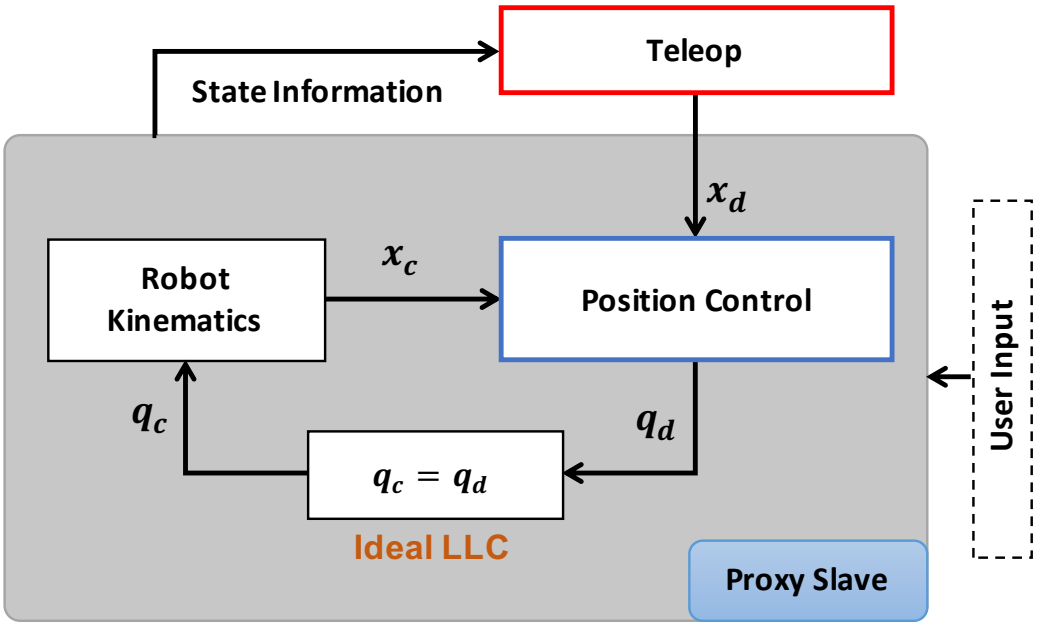


Figure 5.6: Proxy Slave : q_c-current joint state, q_d-desired joint state, x_c-current cartesian state, x_d-desired cartesian state

to a simulated slave robot, called the *Proxy Slave*. The user teleoperates the proxy slave robot and based on the interactions with the situational model, *Teleop* sends appropriate commands to the *Master MLC* and *Slave* MLC. The advantage of the *Proxy Slave* is that it provides the state information of the slave robot with an ideal position control, i.e., ideal *LLC*. This allows the *Teleop* component to use the contact information of the proxy slave robot with the situational model for haptic rendering on the master side. Additionally, *Teleop* uses this information to send admittance commands to the *Slave MLC* for force-limiting and other task-specific constraints.

5.2.1 Motivation

Typically, in an MMT approach, as the model of the task environment is learned, it is fed to the master device to haptically render user feedback. The quality of haptics usually depends on the accuracy of the model and the noise in the contact information. This works perfectly only when the model of the task environment is known and the master device has very accurate gravity compensation module. Further, one of the goals of the CSA framework is to provide adaptability to any robotic platform, ensuring that any master device with haptic capability will be able to use the CSA framework. The above-mentioned reasons motivate the need for a "Proxy Slave". It should be noted that *Proxy Slave* still allows the haptic rendering on the master device, even though the model of the task environment is registered to the proxy slave robot, rather than the master handle. Additionally, from the developer's point of view *Proxy Slave* can be used as simulated slave robot to test various motion primitives before employing them on the real system.

5.2.2 Control Law

This component obtains position commands from the *Teleop* component and executes the motion based on the position controller only. The minimization problem

is similar to that of Equation (5.1):

$$\min_{\delta q} \|J\delta q - (x_d - x_c)\|$$

such that,

$$v_l \leq \frac{\delta q}{\delta t} \leq v_u$$

where, J represents the body jacobian of the proxy slave robot and δq corresponds to incremental joint motion. v_l and v_u correspond to lower and upper joint velocity limits, respectively.

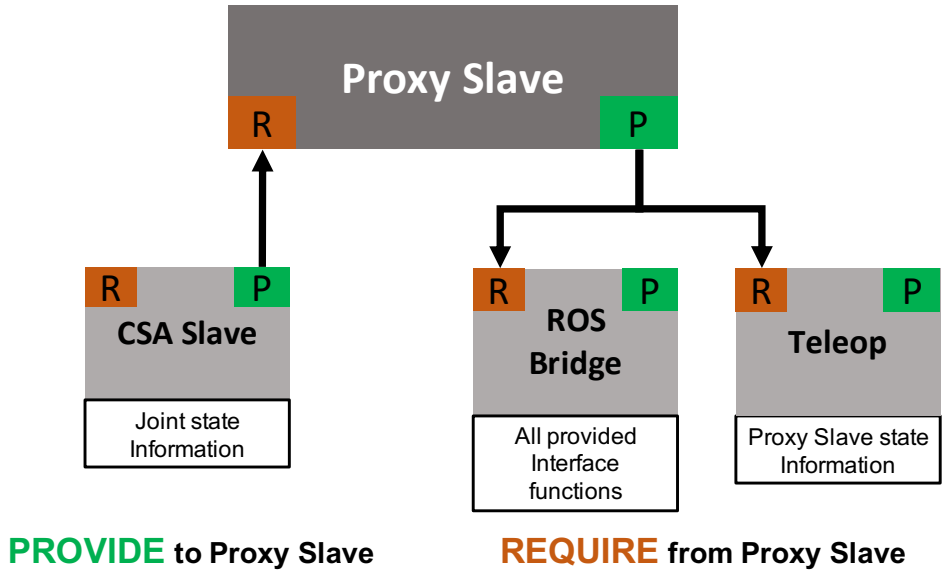


Figure 5.7: *Proxy Slave* interface connections

5.2.3 Interface Connections

Proxy Slave is connected to the provided interface of *CSA Slave* for joint state information for the purpose of intermittent joint synchronization between both of the slave robots. Since the proxy slave robot does not have any low-level control, this allows the proxy slave to re-synchronize its joint state with the actual slave robot during homing or when idle. Further, the *Teleop* connects to the provided interface of the *Proxy Slave* to retrieve robot state information. Figure 5.7 shows the interface connections for the *Proxy Slave*.

5.3 CSA Master

The *CSA Master* process is responsible for the control of the master manipulator. This process consists of two sub-processes: a *Master MLC*, which communicates with the *TeleOp* process, and a *Master LLC*, which communicates with the master manipulator and performs basic joint-level servo control functions. The *Master MLC* runs as a clock-driven process at a repetition rate of approximately 100-500 Hz. The *Master LLC* typically may run at a faster duty cycle. Most of the hardware-dependent components will reside in the *Master LLC*, so it should be possible to adapt the system to use other master manipulators in a fairly straightforward manner.

The *Master MLC* receives impedance specification commands from the *Teleop* process and translates them into an appropriate form for execution by the *Master LLC*.

The *Master MLC* process also returns state information to the *Teleop* process, including positions and velocities in joint and cartesian spaces, gripper state of the master manipulator, and forces and torques exerted by the master robot on the surgeon's hand.

5.3.1 Control Law

Much like *CSA Slave*, *CSA Master* also extends the *dVRK Master*. The *CSA Master* is implemented as a torque controller, same as *dVRK Master*, shown in Figure 5.8. This allows the user to provide external joint torques computed from various control goals. The component is responsible for adding all the necessary joint torques and sending them to the *Master LLC*, which communicates with the master manipulator to perform joint-level servo control.

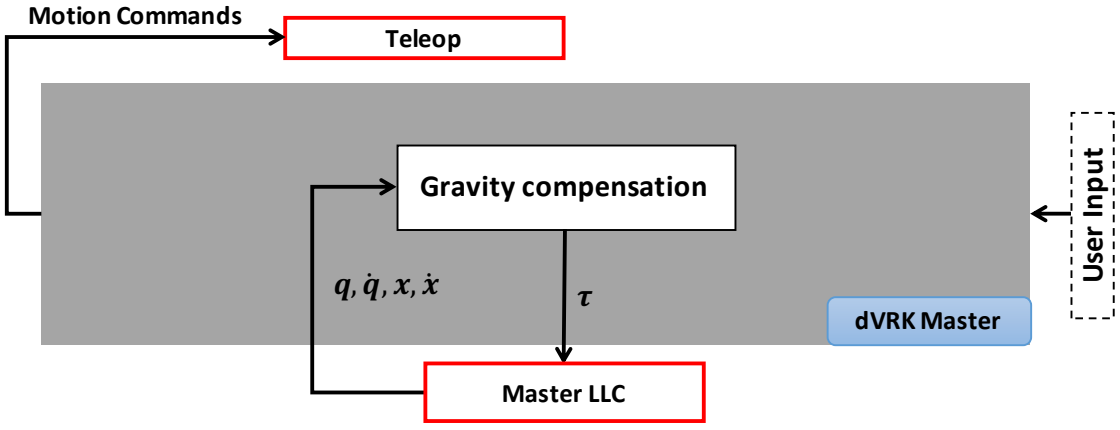


Figure 5.8: dVRK Master : q-joint position, $\dot{q}$-joint velocity, x-cartesian position, $\dot{x}$-cartesian velocity, τ-total joint torque sent to the *Master LLC*

Algorithm 5.1 Compliance wrench

Output: Wrench $[f, t]$

$\vec{\epsilon} = F_c^{-1}\vec{p} = R_c^{-1}(\vec{p} - \vec{p_c})$ ▷ Position Error

$\vec{v} = R_c^{-1}\dot{p}$ ▷ Velocity on compliance frame

for $i \in \{x, y, z\}$ **do**

  if $\vec{\epsilon}_i \leq 0$ **then**

    $\vec{g}_i = \vec{g}_i^{(-)} + \vec{k}_i^{(-)}\vec{\epsilon}_i + \vec{b}_i^{(-)}\vec{v}_i$

  else

    $\vec{g}_i = \vec{g}_i^{(+)} + \vec{k}_i^{(+)}\vec{\epsilon}_i + \vec{b}_i^{(+)}\vec{v}_i$

  end if

end for

$\vec{f} = R_c\vec{g}$ ▷ Desired force

$\vec{\theta} = Rodriguez(\triangle R = R_c^{-1}R)$ ▷ Orientation Error

for $i \in \{x, y, z\}$ **do**

  if $\vec{\theta}_i \leq 0$ **then**

    $\vec{\tau}_i = \vec{\tau}_i^{(-)} + \vec{k}_{oi}^{(-)}\vec{\theta}_i$

  else

    $\vec{\tau}_i = \vec{\tau}_i^{(+)} + \vec{k}_{oi}^{(+)}\vec{\theta}_i$

  end if

end for

$\vec{t} = R_c\vec{\tau}$ ▷ Desired torque

One such goal is compliance control, which requires the user to provide compliance gains to the *Master MLC*. Using these gain parameters, the compliance wrench is computed based on the pseudocode described in Algorithm 5.1. Here, the compliance frame $F_c = [R_c, p_c]$, defined in the base frame of the master robot, is used to calculate the desired wrench $[f, t]$. The position stiffness gains, $\vec{k}^{(+)}$ and $\vec{k}^{(-)}$, the position damping gains, $\vec{b}^{(+)}$ and $\vec{b}^{(-)}$, and the force bias terms, $\vec{g}^{(+)}$ and $\vec{g}^{(-)}$ are used to calculate the desired force. Similarly, torque bias terms, $\vec{\tau}^{(+)}$ and $\vec{\tau}^{(-)}$, and the orientation stiffness gains, $\vec{k}_o^{(+)}$ and $\vec{k}_o^{(-)}$, are used for computing the desired torque. This compliance control is used to provide force feedback to the user on the master

handle, which is helpful for user guidance.

The user also has a provision for sending the compliance wrench directly to the *Master MLC*. This allows for the system to compute the compliance wrench externally and send it to the *Master MLC* for haptic rendering. As shown in Fig. 5.9, the user can either specify the external compliance wrench (f_e, t_e), or provide the compliance gains for the system to compute the compliance wrench (f, t). The compliance torque, τ_c and the torque from gravity compensation, τ_{gc}, are added to get the total desired torque.

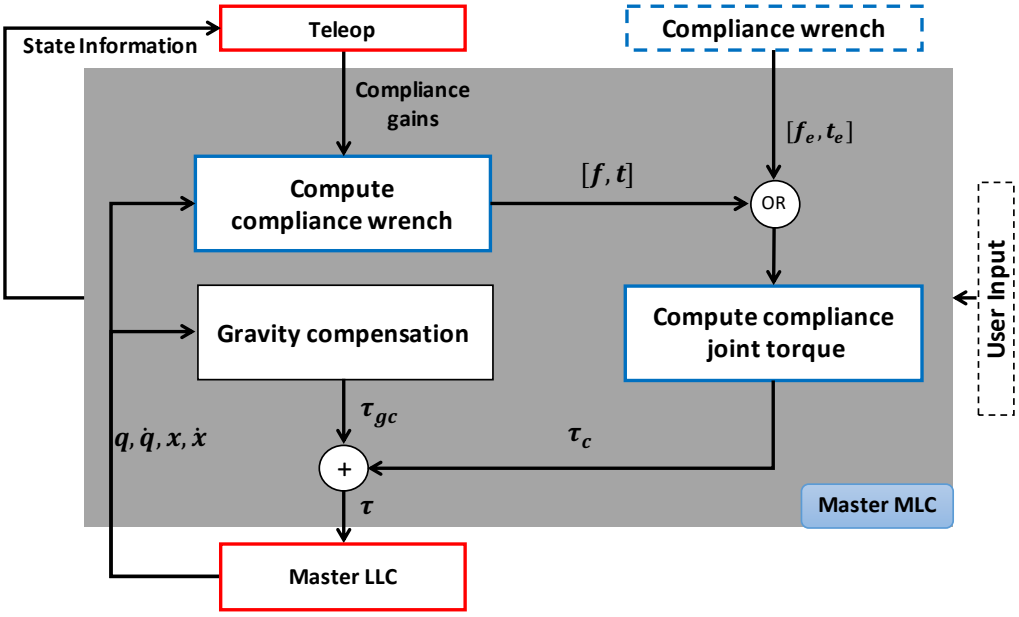


Figure 5.9: *CSA Master* : q-joint position, $\dot{q}$-joint velocity, x-cartesian position, $\dot{x}$-cartesian velocity, $[f_e, t_e]$-compliance wrench from external source, $[f, t]$-wrench computed based on compliance algorithm, τ_{gc}-joint torque from gravity compensation, τ_c-joint torque from compliance control and τ-total joint torque sent to the *Master LLC*

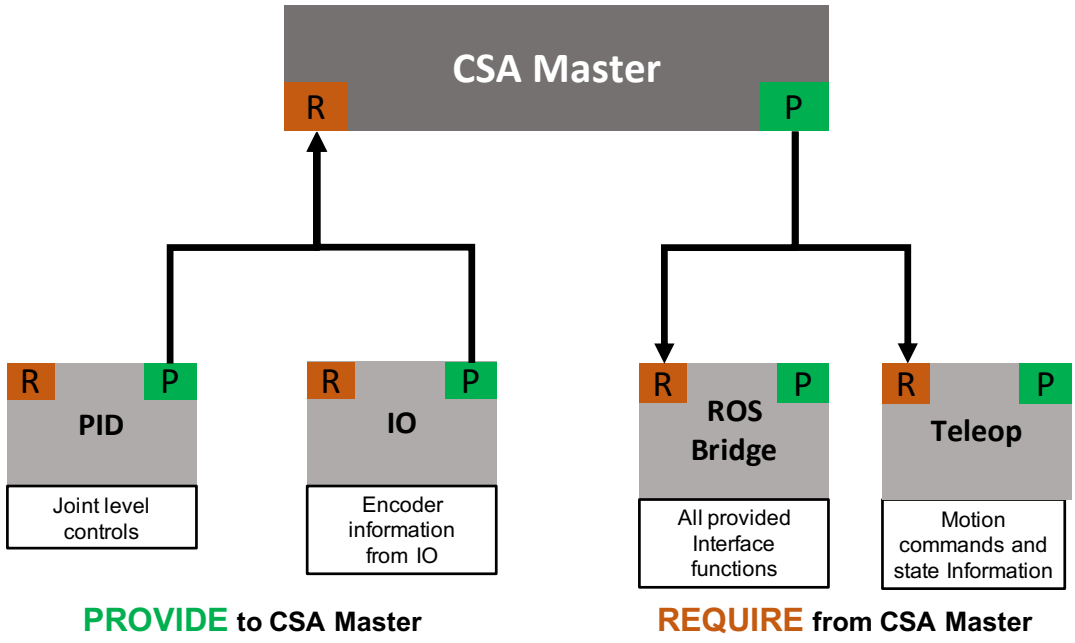


Figure 5.10: *CSA Master* interface connections

5.3.2 Interface Connections

Figure 5.10 shows the interface connections for the *CSA Master*. The component has a provided interface that connects to *ROS Bridge* to export useful functionality to the ROS environment. *Teleop* also connects to the provided interface to retrieve robot motion commands, state information, and functions to populate impedance gains to the torque controller. *CSA Master* connects to the *PID* and *IO* interfaces, which are part of *LLC*, to retrieve joint- and encoder-level information, respectively.

5.4 Contributions

The contributions reported in this chapter include:

1. *CSA Slave* architecture, which extends the *dVRK Slave* with the addition of hybrid force-motion control.

2. Incorporation of various motion primitives for continuous palpation tasks.

3. *CSA Master* architecture, which extends *dVRK Master* with inclusion of compliance control for haptic visualization. This feature is readily available in the latest release of the dVRK for the research community to use.

5.5 Published Work

Material from this chapter has appeared in the following publications:

1. P Chalasani, A Deguet, P Kazanzides, RH Taylor, "A Computational Framework for Complementary Situational Awareness (CSA) in Surgical Assistant Robots," in 2018 Second IEEE International Conference on Robotic Computing (IRC), 9-16

2. P. Chalasani, L. Wang, R. Yasin, N. Simaan, and R. H. Taylor, "Preliminary evaluation of an online estimation method for organ geometry and tissue stiffness," in 2016 IEEE Robotics and Automation Letters, vol. 3, no. 3, pp. 18161823.

3. L. Wang, Z. Chen, P. Chalasani, R. M. Yasin, P. Kazanzides, R. H. Taylor, and N. Simaan, "Force-controlled exploration for updating virtual fixture geometry in model-mediated telemanipulation," in 2017 Journal of Mechanisms and Robotics, vol. 9, no. 2, p. 021010

4. P. Chalasani, L. Wang, R. Roy, N. Simaan, R. H. Taylor, and M. Kobilarov, "Concurrent nonparametric estimation of organ geometry and tissue stiffness using continuous adaptive palpation," in 2016 IEEE International Conference on Robotics and Automation (ICRA), May 2016, pp. 4164-4171

Chapter 6

System-Level Tests

In this chapter, details of various tests are described to assess some of the features implemented in the CSA framework. Table 6.1 shows the various functionalities that were tested/evaluated in this chapter.

	Section			
	6.1	**6.2**	**6.3**	**6.4**
Master-side VF	[F]	[F,E]		
Slave-side VF	[F]			
Palpation	[F]			
Ultrasound Sensing			[F]	
Confocal Endomicroscopy Sensing				[F,E]

Table 6.1: Outline of the chapter. [F] - Feasibility, [E] - Evaluation

6.1 Constrained Teleoperation with Task-Specific Constraints (dVRK)

In this subsection, we present a constrained teleoperation example where the user selects a region of interest to explore. Corresponding boundary virtual fixtures are enabled to allow the user to palpate in the region of interest. In a surgical task, this behavior enables the surgeon to isolate the region of interest where a tumor might be and perform a constrained robotic palpation rather than palpating the entire surface.[i]

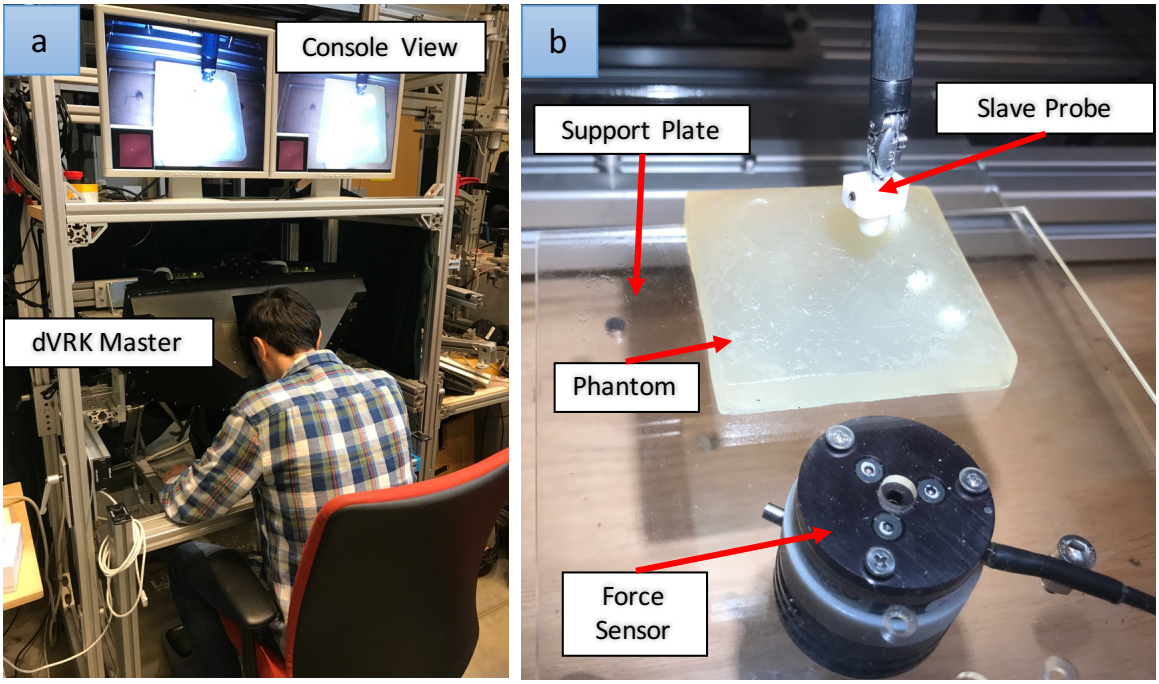


Figure 6.1: a) User teleoperating using the dVRK master device; b) slave side setup with phantom and force-torque sensor.

[i]This experiment was performed at Johns Hopkins University; Anton Deguet volunteered to be the user for the demo.

This test was performed on a silicone phantom with an embedded stiff feature resembling a tumor as shown in Figure 6.1b. An ATI Mini-25 force-torque sensor (ATI Industrial Automation, Apex, NC, USA) was used to measure the interaction forces. In our setup, we placed the force-torque sensor underneath the support plate holding the phantom, however, the system is capable of receiving force information over the network. With a conventional teleoperation system, the user would have to perform discrete probing on the phantom and an offline estimation is done to estimate the tissue stiffness. With the help of CSA framework, the user can either explore the entire surface using the continuous palpation primitives described in Section 5.1.3 or narrow down the exploration to a specific region of interest (ROI). The user needs to provide a few point locations enclosing the region of interest (ROI) using the master gripper. The system then generates a Forbidden Region Virtual Fixture (FRVF), constraining the motion of the slave end effector inside the ROI. The compliance wrench was also calculated based on the interaction of the Proxy Slave with the situational model. The estimated compliance wrench was sent to the *Master LLC* to render force-feedback at the handle of the master console. Figure 6.2 shows the master console display as seen by the user. The situational model is located on the bottom left, which includes the rendering of the ROI (computed using a convex hull) using points selected by the user. The palpation information was then used to generate stiffness information while the user was palpating.

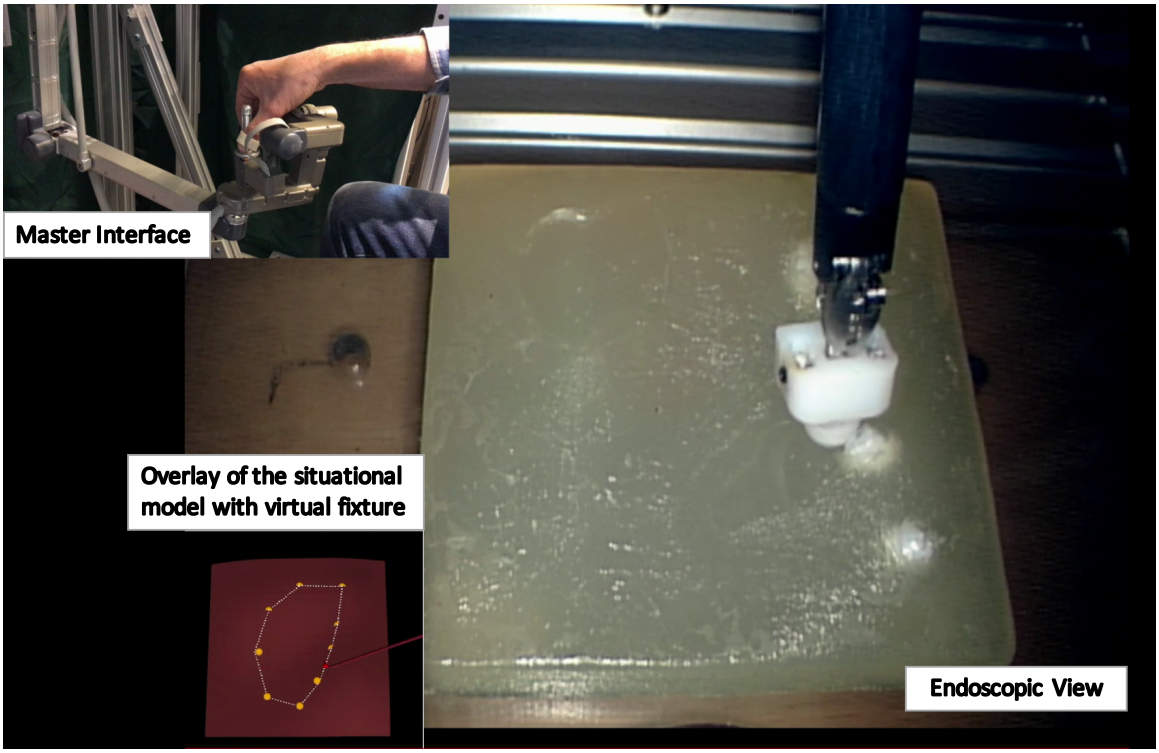


Figure 6.2: Endoscopic view as seen by the user. (Bottom Left) Overlay of the situational model on the console view. (Top Left) This is not an overlay, it is for the reader's benefit to show the user performing constrained teleoperation on the situational model.

6.2 Model-Mediated Teleoperation with Path Following VF (dVRK)

In this subsection, we demonstrate the advantage of MMT to improve the path following performance and to shorten the task completion time. The approach described here directly correlates to the force controlled ablation task. In this experiment, two silicone models made of silicone elastomer (M-F Manufacturing, Fort Worth, Texas) with an embedded stiff feature were used, as shown in Figures 6.3 and 6.4.[ii]

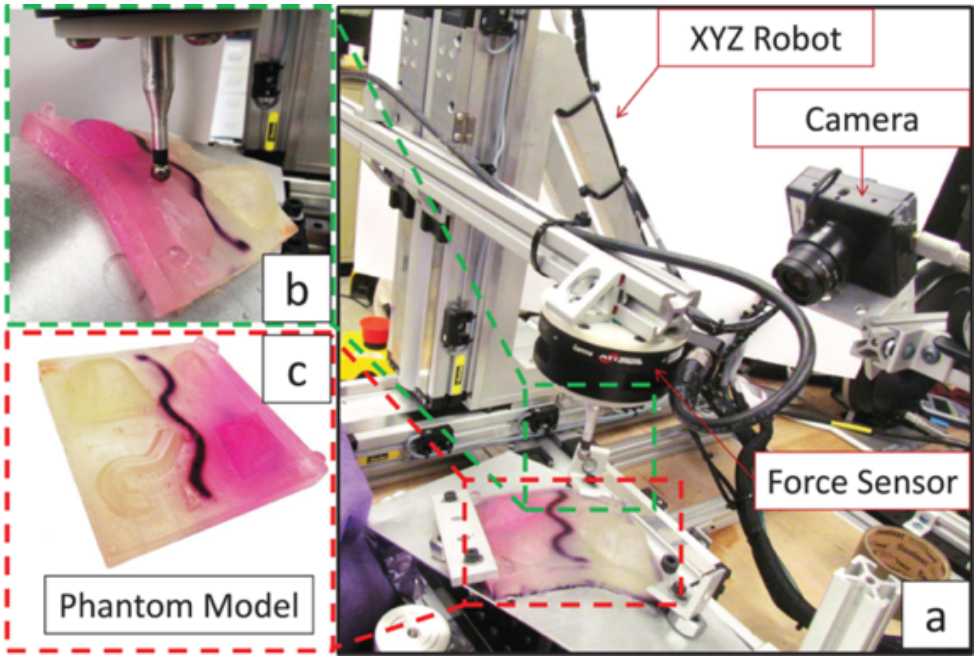


Figure 6.3: Cartesian XYZ robot : (a) experiment setup, (b) ball probe finger ATI force torque sensor, and (c) a phantom model used in the experiment.

[ii]This experiment was performed at Vanderbilt University and Johns Hopkins University; Members include Long Wang, Zihan Chen, Rashid M. Yasin, Peter Kazanzides, Russell H. Taylor and Nabil Simaan. I was responsible for providing the CSA framework and performing the registration. The primary author and others were responsible for conducting the experiment and analyzing the data. Details of this experiment can be found in [105].

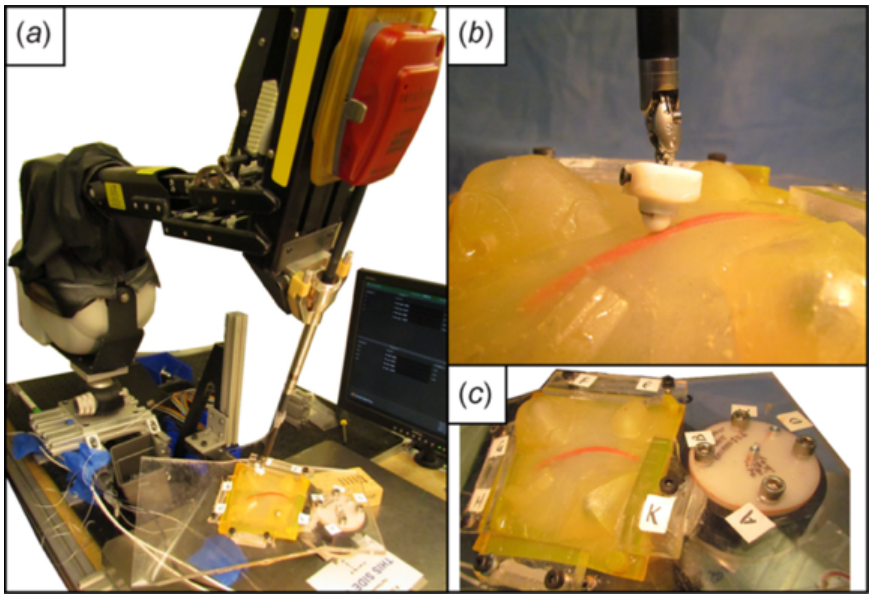


Figure 6.4: dVRK PSM robot: (a) experiment setup, (b) ball probe finger adapter integrated with EM tracker, and (c) a phantom model mounted on a force plate.

An STL file representing an undeformed silicone phantom was obtained from a CAD model using Creo Parametric™. The undeformed silicone model was laser scanned using a Faro Arm FusionVR to generate a point cloud (PC_a). Another point cloud C_a denoting a mockup preoperative plan was also marked on the undeformed model and digitized using the Faro Arm. Figure 6.5 shows the digitized a priori point cloud PC_a along with the curve data C_a.

For the path following virtual fixture, we need to register the curve point cloud C_a to the current deformed model. To gather point cloud (PC_d) of the deformed model, we performed surface exploration using the Cartesian XYZ stage and also using the dVRK PSM. Figure 6.6 shows the exploration data collected using the Cartesian robot and Figure 6.7 shows the exploration data collected using the dVRK PSM.

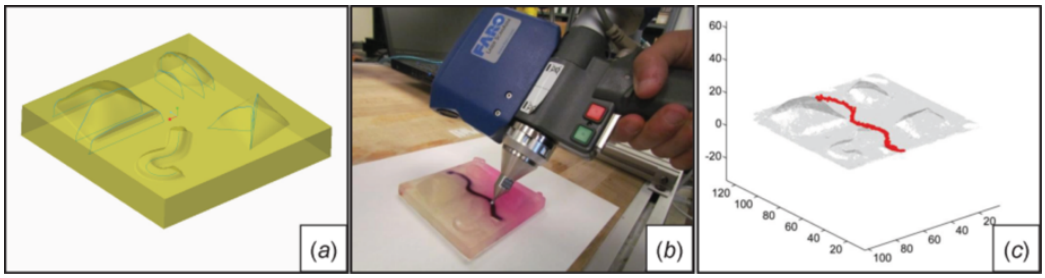


Figure 6.5: Creating an a priori model of the silicone phantom: (a) a priori STL model, (b) digitizing the target curve using Faro Arm, and (c) a priori point cloud with curve information.

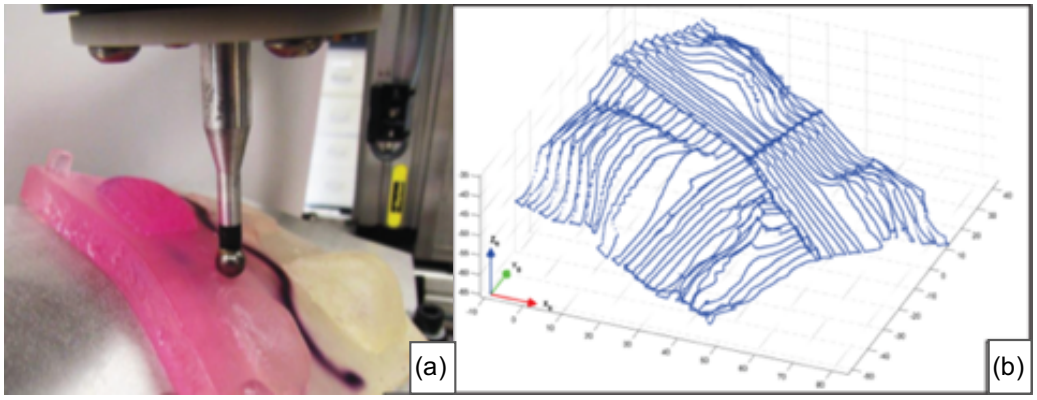


Figure 6.6: Force-controlled exploration using the Cartesian robot: (a) Phantom used (b) scanned point cloud

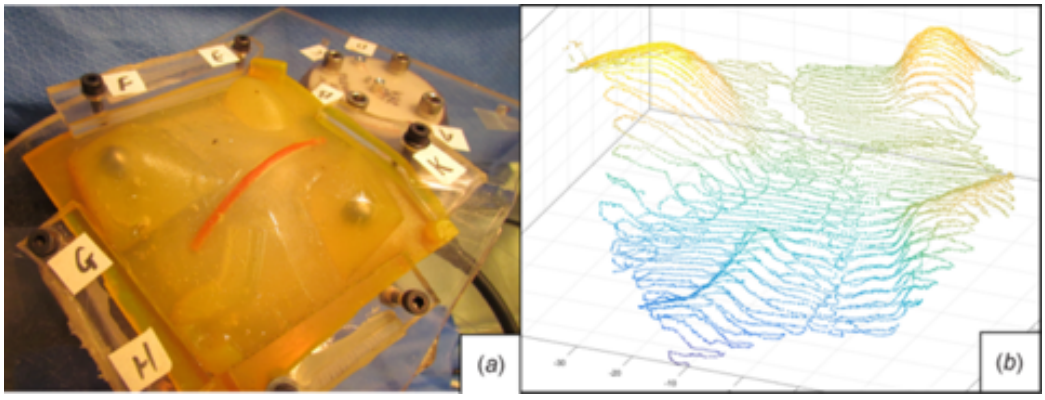


Figure 6.7: Force-controlled exploration using the dVRK PSM : (a) Phantom used (b) scanned point cloud

CPD non-rigid registration was used to deform the a priori point cloud PC_a resulting in $T(PC_a)$, where $T(X)$ represents the transformed point cloud of X. Using the registration information, a deformed curve point cloud $T(C_a)$ was generated from the undeformed curve point cloud C_a. A following virtual fixture was then enabled to guide the user on a preplanned trajectory. Figure 6.8 illustrates the deformable registration process where the blue point cloud represents the exploration data and the red point cloud represents the updated a priori model at a different iteration. Similarly, the green point cloud represents the updated curve at each iteration.

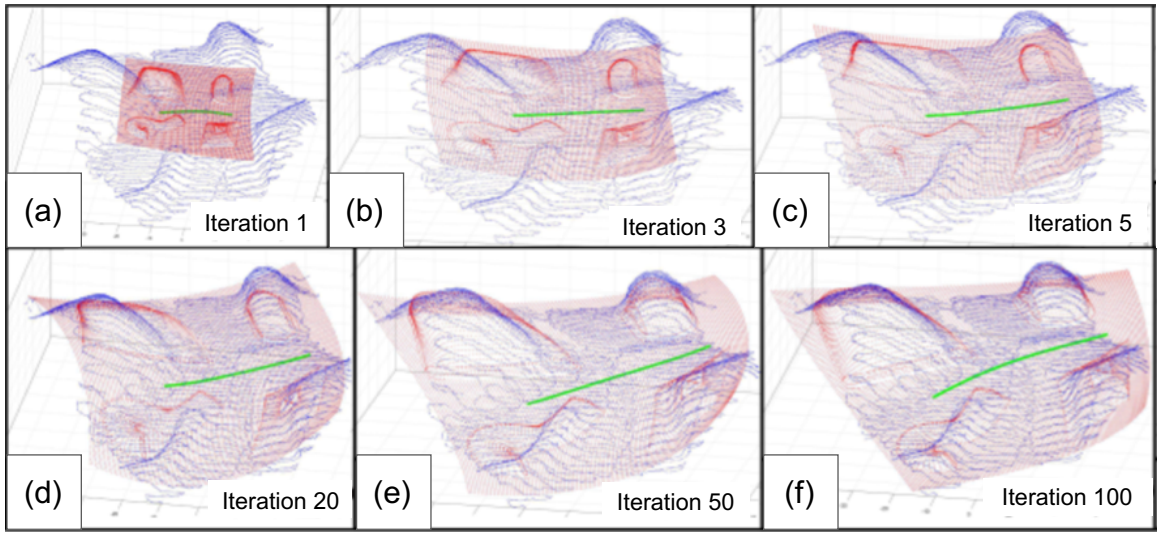


Figure 6.8: CPD result on dVRK PSM exploration data: (a)(f) show result at various iterations of deformable registration using PSM robot data. Blue point cloud corresponds to the exploration data and the red point cloud corresponds to the updated a priori model at a different iterations

To evaluate the performance of the exploration and registration, the actual curve on the deformed model was also digitized to obtain a fitted ground truth C_{gt}. The curve registration error between $T(C_a)$ and C_{gt} is represented by ϵ_o. Table 6.2 shows

the registration results using the Cartesian robot and the dVRK PSM robot where the explorations were performed on the phantoms shown in Figures 6.3 and 6.4, respectively.

Curve TRE	Cartesian	PSM
ϵ_o (mm)	3.393	3.386

Table 6.2: Deformable registration results for the Cartesian robot and dVRK PSM.

Evaluation of the Ablation Task

We also performed feasibility tests to evaluate the use of such updated curve in an assistive VF in which lateral deviation from the desired path is resisted by applying corrective impedance force on the master handle. The Cartesian robot was used with the dVRK master manipulator for experimental validation. Three users participated in the experiment; one was experienced and the other two were not. Each user was instructed to follow the curve with and without the impedance virtual fixture. In the former case, users had to follow the target curve back and forth twice. In the latter case, the impedance virtual fixture was employed to guide the users to follow the curve. Hybrid force motion control was enabled on the slave side in both the cases with a desired force of 0.7N normal to the surface. Visualization on the slave side environment was provided on the master console. The trajectory of the robot tip was recorded along with the task completion time in both the cases. The RMS error and

the completion time of each trial by every user is shown in Tables 6.3 and Table 6.4, respectively. These results show that all users benefited from reduced time for each trial and increased tracking accuracy when the curve-following VF was enabled.

Trial RMS Error						
	Without Virtual Fixture			With Virtual Fixture		
Trial	User 1	User 2	User 3	User 1	User 2	User 3
1	5.40	5.87	5.06	5.06	4.51	6.37
2	4.98	5.63	5.13	5.13	4.85	4.21
3	5.32	5.30	5.00	5.00	4.47	4.40
4	5.22	5.56	4.63	4.63	4.51	4.32
5	5.11	4.85	4.73	4.73	4.71	4.42
Average	**5.21**	**5.44**	**4.91**	**4.91**	**4.61**	**4.75**

Table 6.3: RMS target curve tracking errors for each user (subject) with and without virtual fixture assistance

Trial Completion Time (s)						
	Without Virtual Fixture			With Virtual Fixture		
Trial	User 1	User 2	User 3	User 1	User 2	User 3
1	18.21	22.53	36.24	17.37	14.19	28.81
2	15.68	19.37	38.01	17.20	9.32	20.67
3	13.65	15.34	32.21	15.13	11.47	25.79
4	12.14	17.33	38.69	14.52	10.39	27.12
5	11.90	14.86	29.69	10.94	10.58	26.93
Average	**14.32**	**17.89**	**34.97**	**15.03**	**11.19**	**25.86**

Table 6.4: Trial completion time for each user (subject) with and without virtual fixture assistance

6.3 Teleoperated Ultrasound Scanning (UR3)

In this section, we demonstrate the use of ultrasound sensing modality to locate and segment stiff inclusions in an organ, using the B-mode image. This information is then used to update the preoperative Computed Tomography (CT) image. This is to show the feasibility of performing a teleoperated ultrasound scanning with minimal tool-tissue forces. Using the intraoperative ultrasound data, the model of the task environment is updated.[iii]

Figure 6.9: UR3 robot with an ultrasound transducer attached to the end-effector

[iii] This experiment was conducted at Johns Hopkins University; I was responsible for integrating the CSA framework with the UR control, to perform force-controlled ultrasound scanning. Other team members, Baichuan Jiang, and Zhaoshuo Li were responsible for conducting the calibration, segmentation, and registration. Text for Sections 6.3.1, 6.3.2 and 6.3.3 was provided by Baichuan Jiang.

For this experiment, we first deployed the CSA framework on the UR3 slave arm. We attached a linear probe (Ultrasonix L14-5W) to the end-effector of the UR3, as shown in Figure 6.9. The UR3 was teleoperated using the dVRK master and followed the hybrid force-motion control, defined in Section 5.1.2. The calibration procedure that is used to register the linear probe with the UR3 is described in Section 6.3.1. Later, we perform image segmentation and registration, the procedure of which is described in Sections 6.3.2 and 6.3.3, respectively.

6.3.1 Calibration

To obtain the 3D information from a 2D ultrasound probe, it is required to have some form of tracking capability and establish the transformation relationship between the tracker and ultrasound image. The process of computing this transformation is called ultrasound calibration, regardless of the tracking modality. In our setup, we are using the UR3 robot kinematics to track the 3D information during ultrasound scanning, and BXp calibration method,[128,129] as shown in Figure 6.10, is used to identify the transformation from robot end-effector to ultrasound images.

We have a cross-wire phantom that is composed of two crossing fishing lines. After installing the cross-wire phantom in the water tank, the crossing point can be identified in the ultrasound image. By moving the robot to different poses that capture the physically-same cross-wire point in the ultrasound image, a set of robot frames B_i and a set of fiducial point locations in the images p_i are collected. Then,

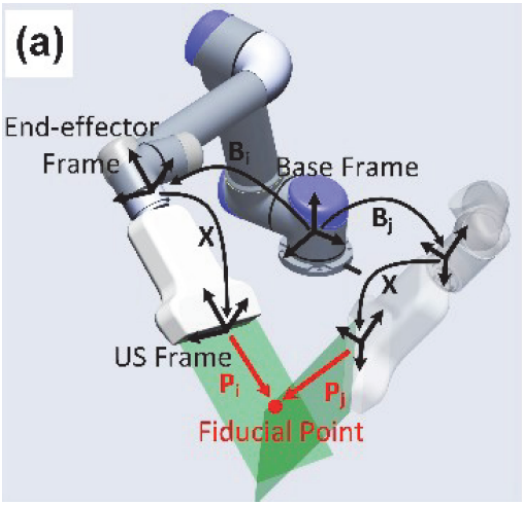


Figure 6.10: BXp type ultrasound calibration[128]

we can establish the set of equations below (where c denotes the constant fiducial point physical location):

$$B_i X p_i = c \tag{6.1}$$

Plugging in the data B_i and p_i and solving for X is essentially equivalent to finding the set of parameters, $param = [Xx, Xy, Xz, Xrx, Xry, Xrz, cx, cy, cz]$ that minimizes the equation:

$$\min_{param} (B_i X p_i - c) \tag{6.2}$$

where $[Xx, Xy, Xz, Xrx, Xry, Xrz]$ denotes the 6DOF parameters of X and $[cx, cy, cz]$ denotes the 3D component of c. After solving for X, the cross-wire is moved to a different physical location, and 5 more data points are collected to validate the computed X. The result is shown in Figure 6.11.

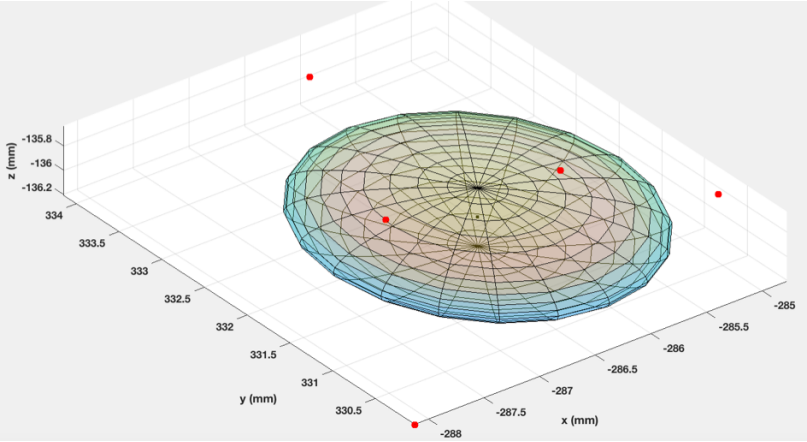


Figure 6.11: Bxp Result: Red points represent the computed cross-wire physical locations in the robot base. Black point at the center represents the average of all red points. Ellipsoids represents the standard deviation of 1.23mm, 1.54mm and 0.24mm in x, y and z direction, respectively.

6.3.2 Segmentation

The features that we are going to extract are the lesion boundaries and the vessel boundaries, both of which are shown hypoechoic in the ultrasound images. Therefore, we are going to use the Active Contours algorithm (Snakes) introduced in [130] to extract the boundary features. In general, the active contours algorithm utilizes a "snake", which is an energy minimizing, deformable spline influenced by constraints and image forces that pull it towards object contours and internal forces that maintain smoothness and resist deformation. Although the active contours algorithm can achieve the degree of accuracy and speed that the project requires, it heavily depends on a good initialization. To continuously extract boundaries from a sequence of ultrasound images, the following initialization scheme is proposed:

- The image is first smoothed by a median filter (Figure 6.12a).

- Then a threshold filtering is applied to the smoothe anything else to say main menu hanger thank you for Comcast image, which produces the image mask A (Figure 6.12b)
- An optional process is doing an open-close operation on mask A to remove smaller islands
- Then the mask B is created which is based on the ultrasound imaging setting, i.e., imaging depth and probe width (Figure 6.12c)
- We do an AND operation on mask A and B so that only the region of interest (ROI) is used for initializing the mask
- Finally, we use the combined mask as the initialization for the active contours algorithm and the result is shown in (Figure 6.12d).

Because the ultrasound image sequence is time- and space- consecutive, the segmentation result (boundaries extracted) is directly fed into the next slice and used as the initialization mask. This will speed up the processing time and reject temporary imaging artifacts. However, if we always use the previous result as the mask, some new isolated features will not be captured by the active contours algorithm. Therefore, we do the initialization every 10 image frames, which is a trade-off between efficiency and robustness.

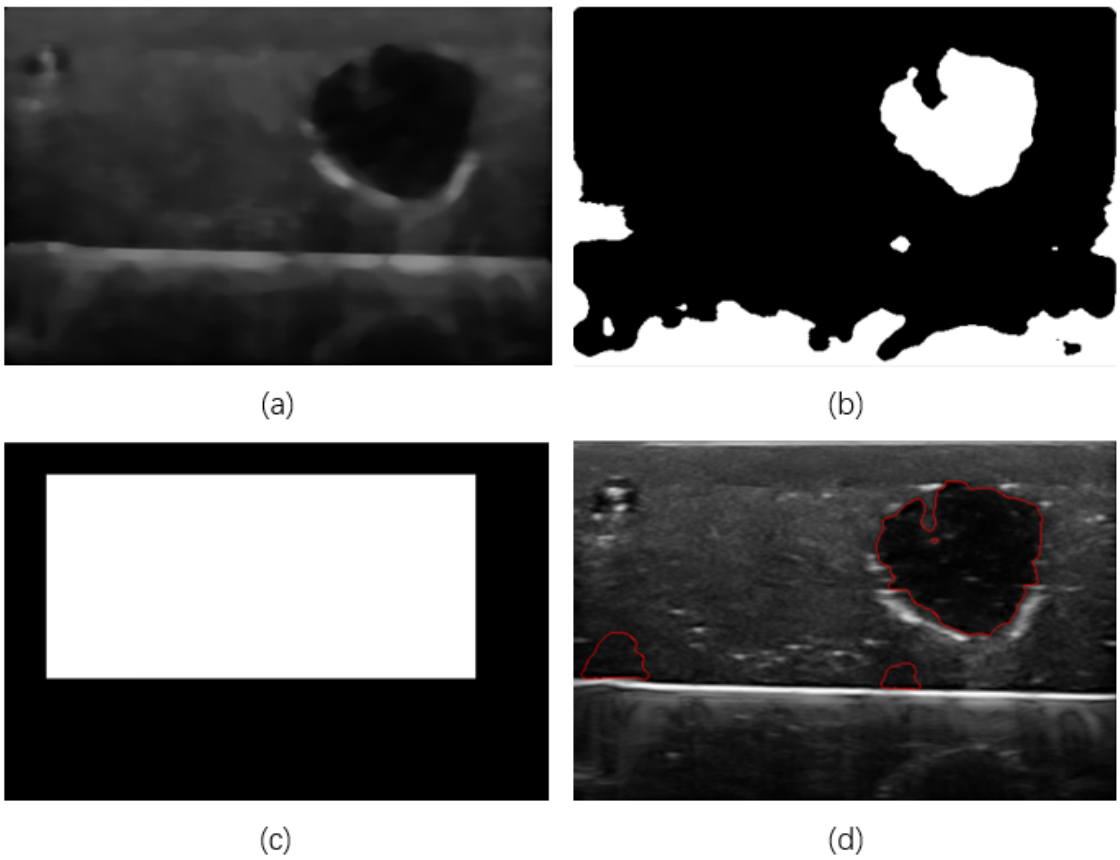


Figure 6.12: Mask initialization and segmentation results, a) Result of a median filter; b) Result after applying threshold filtering to the smoothened image; c) Mask based on image depth and probe width, and d) Result of active contours algorithm

6.3.3 CT Registration

We have done a Cone-Beam Computed Tomography (CBCT) scan of the kidney phantom, which is used as the ground-truth preoperative data for performing the registration with the intraoperative ultrasound features. The internal lesions and vessel in the kidney phantom are manually segmented and shown in Figure 6.13.

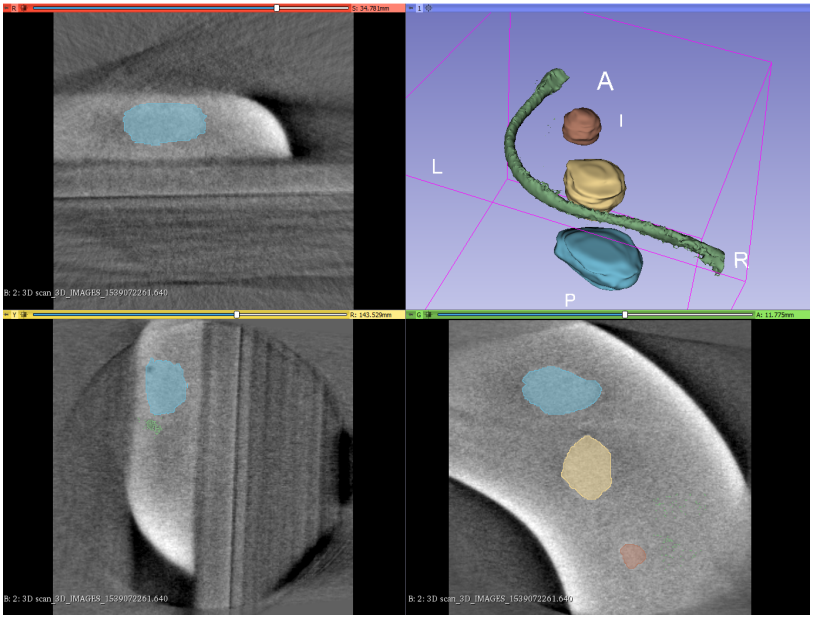


Figure 6.13: CBCT scan of the kidney phantom with the internal feature segmentation

After obtaining the ultrasound features, we aggregated and downsampled the extracted contour lines to form a point cloud as shown in Figure 6.14. We used the CPD registration algorithm, reported in [51], which resulted in a registration error of 7.9303 mm, as shown in Figure 6.15.

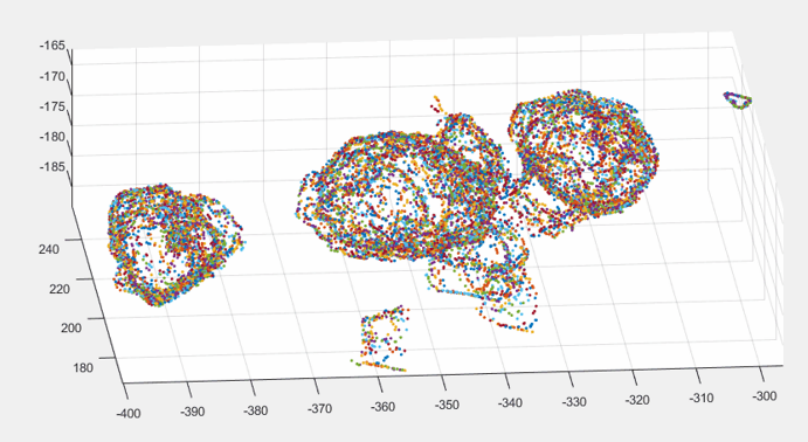


Figure 6.14: Contour points extracted from ultrasound image sequence

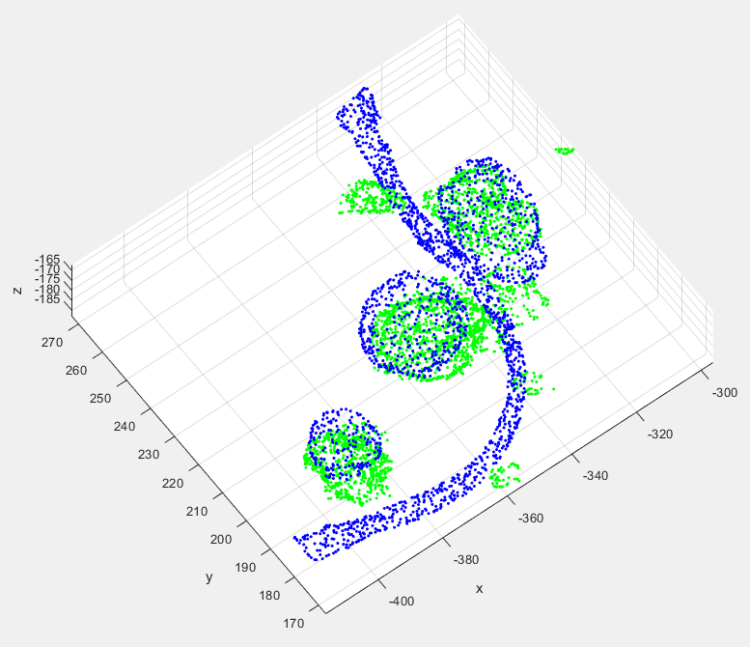


Figure 6.15: Registration result. Blue: CT point cloud. Green: Ultrasound point cloud

6.3.4 Discussion

This experiment demonstrated the feasibility of using robotic ultrasound sensing for updating the preoperative CT model using the intraoperative B-Mode information. We have further validated the CSA platform by adapting the framework on to the UR3 arm, that was teleoperated by the dVRK master robot. However, there are some limitations that can be addressed in later iterations of the framework. The active contours algorithm, used for segmentation, is sensitive to the mask initialization. This caused some image artifacts, which affected the registration algorithm. Therefore, the CPD algorithm produced a larger registration error than expected, even though it was tuned for outlier rejection.

In the future, these limitations can be addressed by having a better segmentation algorithms as described in [131–133]. These can be further modified to perform in an online manner, which in turn would complement the registration process to perform as fast as possible.

6.4 Confocal Endomicroscopy (dVRK)

Recent advances in optical biopsy techniques, and, in particular, probe-based confocal laser endomicroscopy (probe-based Confocal Laser Endomicroscopy (pCLE)), have demonstrated great promise for real-time in vivo tissue characterization. Mosaicking algorithms are often required to provide macro coverage of the tissue surface due to limited field-of-view of the current pCLE probes. Acquisition of high-quality contiguous image streams is extremely challenging when acquired manually. With the help of our colleagues at the Hamlyn Center for Robotic Surgery, Imperial College London we developed a real-time closed-loop controller for the axial probe position based on image quality metric alone. This sensorless framework integrated with CSA allowed us to perform real-time autonomous probe-tissue contact management during pCLE scanning.[iv]

The experiment was conducted at the Hamlyn Center and the hardware setup comprises of two main systems: a) dVRK master and slave manipulators with integrated CSA framework, and b) a line-scan confocal laser endomicroscopy system built by the Hamlyn Center. The dVRK controllers are connected in a daisy chain topology and communicate with the host PC via an IEEE 1394 firewire connection cable. A detailed description of the dVRK system is provided by Kazanzides et al. [43]. The

[iv] This experiment was conducted at Hamlyn Center with remote assistance from Johns Hopkins University; Members include Eimear O' Sullivan, Lin Zhang, Khushi Vyas, Russell H. Taylor, and Guang-Zhong Yang. I was in charge of providing the CSA framework to test their blur-to-motion algorithm. The text was adapted from an unpublished manuscript primarily co-authored by Eimear O' Sullivan and me. This is expected to be submitted later to a journal article.

laparoscope was held by the da Vinci Endoscope Camera Manipulator (ECM) arm and provides 720x576 SD PAL video streaming for left and right camera channels at a rate of 25Hz. The stereo video stream was captured on the host PC using a Blackmagic (Victoria, Australia) DeckLink Duo 2 PCIe video capture card. A high-speed laser line-scan confocal laser endomicroscopy system (built by Hamlyn Center) was used for image acquisition and validation of the developed framework. The system comprises of a line-scan confocal microscope coupled to a Cellvizio GastroFlex UHD Probe (Mauna Kea Technologies, Paris, France) 30,000 core fiber-bundle with distal micro-lens and a maximum outer diameter of 2.6mm. The system uses a 488 nm laser source and provides a Field of view (FOV) of 240μm and fiber-limited resolution of approximately 1.5μm at 120 frames per second. A detailed description can be found at [134]. Endomicroscopy images were pre-processed by a LabVIEW program, scaled to 300x300 pixels and transmitted via TCP/IP connection to the dVRK system controller.

The software component of the developed framework runs on ROS. The laparoscope images enable the estimation of the pose and position of the probe in the camera frame using a KeyDot marker (KeyDot, Key Surgical, Minnesota) attached to the probe adapter (Fig. 6.16a). The stereo camera and capture card were calibrated using the method outlined in [135] to extract the intrinsic and extrinsic camera parameters. Fast Fourier transform based Normalized Cross-Correlation (NCC) [136] was used to provide real-time mosaic synthesis. Consecutive images from the endomicroscope

data stream were assessed to determine the point of maximum image correlation, providing a robust rotation invariant approach to composing the desired contiguous image mosaic. All experiments were conducted using lens paper stained with 0.2% of acriflavine hydrochloride solution.

Note: Complete algorithmic details of image classification and the blur calculation are not provided in this document. Further details of this work can be found in [137].

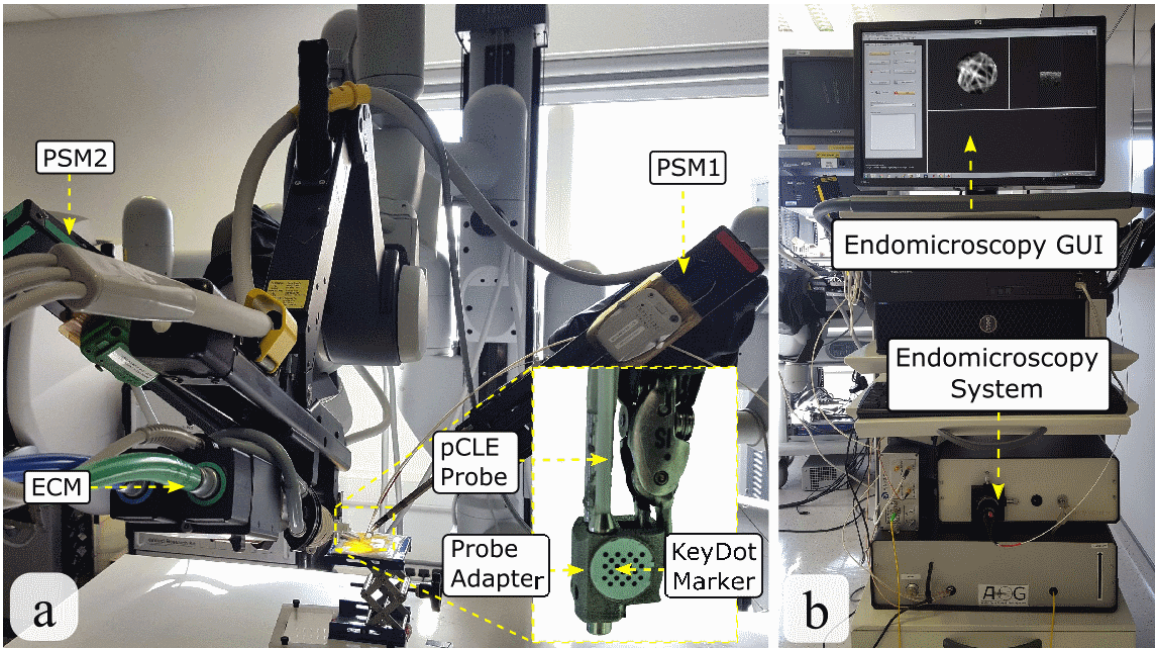


Figure 6.16: System Overview at the Hamlyn Center. a) The da Vinci surgical robot system and a zoomed in view of the pick-up adapter for pCLE and OCT; b) In-house laser line-scan endomicroscopy system.

Teleoperation Framework

The teleoperation framework used in this experiment is based on the CSA framework. Just to recap, CSA supports force-controlled teleoperation, where the slave robot is operated under hybrid force-motion control. The force controller regulates the motion in the direction normal to the surface and the motion controller regulates the motion tangential to the surface. The hybrid control is implemented as an optimization problem to estimate the incremental joint motion. A brief summary of the optimization setup is stated below,

$$\delta q = min \| J\delta q - \delta x \|$$

such that,

$$\delta x = K_a K_g (F_c - F_d)\delta t + K_p(x_d - x_c)$$
$$v_l \leq \frac{\delta q}{\delta t} \leq v_u$$

where J represents the body jacobian of the slave robot, δt is the sampling rate of the component in seconds and δx and δq represent incremental cartesian and joint position, respectively. K_p and K_a are motion and force projection matrices, respectively. K_a projects the motion from the force controller in the direction normal to the surface. Similarly, K_p projects motion from the position controller in the directions

tangential to the surface normal. F_c denotes the contact force obtained from the force sensor and F_d denotes the desired force the slave should maintain with the anatomy. K_g is the admittance gain matrix for the force motion. x_c is the current cartesian position and x_d is the commanded/desired cartesian position. Complete details on the motion control are described in Section 5.1.

In the above control equation, $(F_c - F_d)$ corresponds to a resultant force direction based on the contact force sensed by an external force sensor and a user-specified desired/bias force. In lieu of an external force sensor, the motion control directly receives a motion direction and a magnitude based on image quality using blur-to-motion estimation[v]. The modified equation for calculating the incremental cartesian position (δx) is as follows,

$$\delta x = K_a K_g \Delta F \delta t + K_p(x_d - x_c)$$

where ΔF is the directional motion vector provided by the blur-to-motion algorithm.

Results

We evaluated our teleoperation framework in two different scenarios: i) unconstrained teleoperation without compensation and b) constrained teleoperation where the axial probe motion is regulated by the blur-to-motion control. In both cases, the

[v] the blur-to-motion algorithm is developed by the Hamlyn Center and provides a motion in the axial direction to provide better image quality (analogous to auto-focus) based on the image blur.

rotation of the end-effector was locked to align the probe with the normal direction of the tissue surface.

A Burster 8438-5005 load cell and accompanying micro-controller (Cypress Semiconductor, San Jose, CA) were used for validation purposes to record the forces encountered by the probe throughout the scanning process. The load cell had a range of $0g$ to $500g$ and a linear fitting was derived to convert these values to mN. A windowing filter was applied to the acquired signal to remove excess noise.

Three trials were conducted for both constrained and unconstrained teleoperated scanning. Each trial commenced with the probe a similar distance from the tissue surface and lasted approximately 20 seconds. Data analysis began when the probe first made contact with the tissue surface.

Table 6.5 shows the mean and SD of the forces experienced by the probe for constrained and unconstrained teleoperation during the trials. It can be observed that the recorded values are lower in both cases, indicating that probe-tissue contact forces remained consistently lower when the axial motion was controlled by CSA along with motion feedback from the image blur.

We also observed that the mosaic quality was consistent during constrained teleoperation as shown in Fig. 6.17b. However, the mosaic reconstruction was interrupted during unconstrained scanning due to constant contact loss with the tissue. The interruption can be clearly seen in Fig. 6.17a.

	Unconstrained (mN)	**Constrained (mN)**
Trial 1	18.656 ± 13.764	5.404 ± 1.738
Trial 2	9.729 ± 14.384	6.159 ± 1.647
Trial 3	39.996 ± 47.433	7.208 ± 1.394
Average	$\mathbf{22.699 \pm 25.197}$	$\mathbf{6.257 \pm 1.593}$

Table 6.5: Force values for unconstrained/constrained teleoperation trials on static tissue (Mean ± Sd)

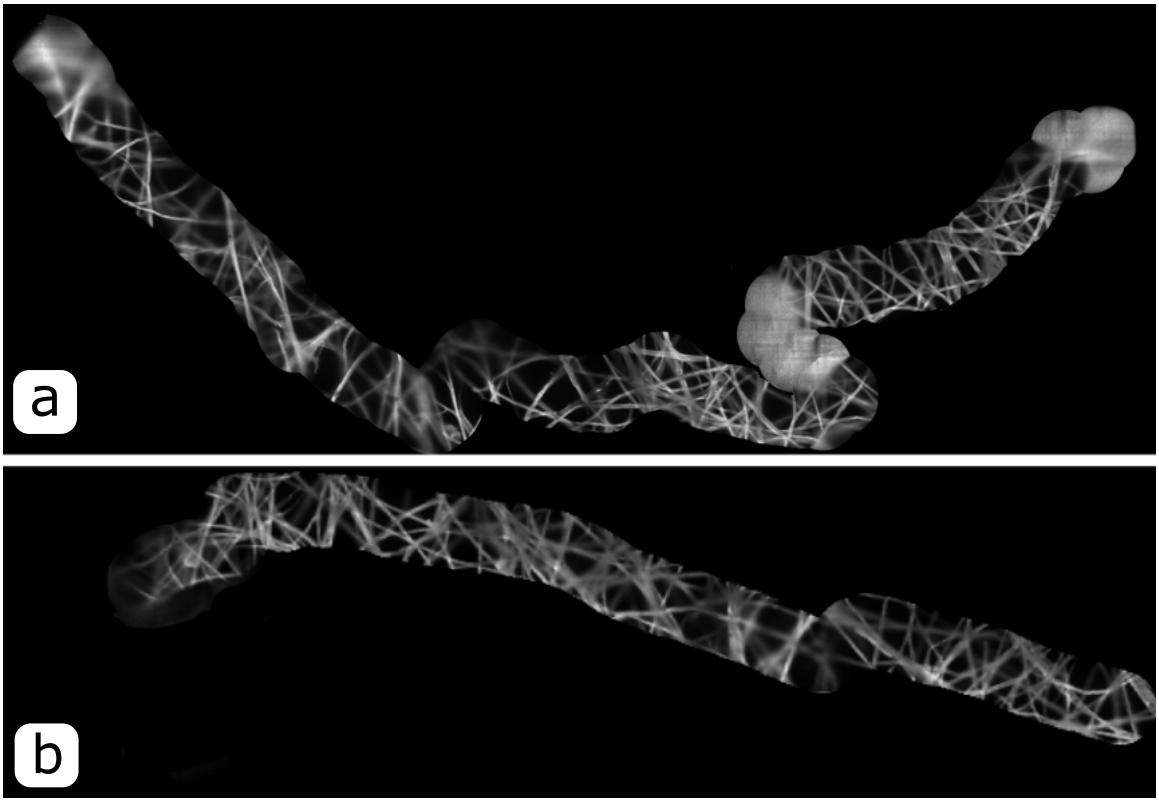


Figure 6.17: a) Mosaic reconstruction for unconstrained teleoperation, b) Mosaic reconstruction for constrained teleoperation with image compensation.

Chapter 7

Conclusions

This dissertation presented a high-fidelity framework that provides real-time situational awareness, for the surgeon, during complex MIS procedures. The proposed framework is called Complementary Situational Awareness (CSA) and provides many machine capabilities, that include sensor information fusion, provision of sensory feedback of the situational model environment, and real-time estimation of the intraoperative model. Developing such a system to provide real-time situational awareness required that many technical challenges be met. These challenges were discussed and addressed throughout the document. The CSA framework has support for multiple sensing modalities. To demonstrate the feasibility, we incorporated the pCLE technique and ultrasound sensing. Further, task-specific assistance is also available in the form of virtual fixtures, along with visual and haptic feedback. The system is developed in a modular fashion and can be incorporated into any existing robotic platform

with minimal overhead. Appendix C also presents a socket based communication to control slave arms that are on different operating platforms.

This chapter provides the summary of all the significant contributions presented in each chapter and discusses possible future work to further improve the CSA framework.

7.1 Summary

Chapter 1 provides introductory material explaining the technical background of the reported work, a succinct statement of the overall thesis of the work, together with a listing of the intellectual contributions reported.

Chapter 2 discusses the various clinical applications that motivated the development of the CSA framework. A surgical workflow is also illustrated, that can be benefited from using the CSA framework. The CSA was designed as a component-based architecture and overviews of various key components are described. The CSA architecture was initially prototyped using the open source dVRK research platform, thus, some key library dependencies are detailed for the reader's comprehension.

Chapter 3 describes the first high-level controller, the *Modeler*. The sole purpose of the *Modeler* is to make sure that the model of the task environment is up-to-date. When the slave robot interacts with the task environment, the contact information is sent to the *Modeler*. The *Modeler* then uses the position-force pairs to estimate the

organ geometry and surface stiffness. We developed a novel technique to estimate this information using the GP. First, tool-tissue forces underneath the organ surface are modeled using the GP, then, using this model surface stiffness and geometry are estimated based on a linear stiffness model. This offline technique was later modified to perform in real-time, by making use of spatial grids. The goal is to store all the contact information but train and predict the organ geometry and stiffness based on neighboring information, rather than the entire surface. This estimation produces fast updates to the geometry and stiffness map which is displayed for the user at interactive frame rates. The estimated geometry and stiffness is then sent to the registration component which performs a rigid, followed by a non-rigid registration technique. The rigid registration is done to correct the initial misalignment as most non-rigid registration algorithms are sensitive to large initial misalignment. For demonstration purposes, we used the rigid registration algorithm, IMLP,[96] and for the non-rigid registration, we used CPD.[51] Based on the estimated stiffness, the user has an option to use the trajectory optimizer to guide the palpation/exploration towards unexplored, but potentially stiff, regions.

Chapter 4 describes the second high-level controller, the *Teleop*. The *Teleop* component maintains the state information of the three MLCs and is responsible for managing communications between those components. The *Teleop* incorporates the MMT paradigm, in which the master robot operates on the model of the task environment by telemanipulating a proxy slave device. Based on the interactions between

the proxy slave device and the model, necessary commands are sent to the real slave control. Further, the master control also receives necessary gain parameters from the *Teleop* for haptic feedback.

Chapter 5 describes the three MLCs, the *Master MLC*, the *Slave MLC*, and the *Proxy Slave MLC*. The *Master MLC* operates on torque control. Apart from gravity compensation, compliance control is also incorporated for visualizing haptics. The estimation of the compliance wrench is also explained based on the interactions of the proxy slave with the model of the task environment. The *Proxy Slave* is implemented as an ideal position control, thus, it is allowed to penetrate the surface of the model in the virtual environment, and the compliance wrench is calculated based on the penetration. The *Slave MLC*, on the other hand, follows hybrid force-motion control. This prevents the slave robot from penetrating the surface of the organ and allows it to servo at a desired force. The mismatch between the model and the reality that occurs when both the slave robots interact with their respective environments is monitored and corrected using different methods explained in this chapter.

Chapter 6 describes the various system-level tests that were conducted to validate different components, and to evaluate different features and algorithms implemented in the CSA framework.

7.2 Future Work

There are many areas in this dissertation where there is room for improvement. In terms of MMT, a more comprehensive user experience can be provided. In the current implementation, the stiffness map and organ geometry is provided as a picture-in-picture display. However, the video feed of the surgical site can be augmented with the tissue properties. For this, a camera calibration needs to be done to register the stereoscopic view to the robot. Many groups have developed different ways to perform the calibration,[138–141] which can be incorporated into the CSA workflow.

In the current GP implementation to estimate the organ geometry and tissue stiffness, we use the spatial grids to store all the palpation position-force pairs and only use the neighboring information during training and prediction. However, this can be further optimized by having localized GP models for each spatial bucket. For prediction, all the neighboring GP models will estimate a predictive distribution and a weighted average of all these predicted means and variances would be the final result. Further, we can implement ways to update the covariance matrix incrementally using methods described in [142–144]. In terms of data pruning, the incoming position-force pairs can be added/removed based on the model entropy.[111,112]

Online estimation of organ properties can also be extended to use/estimate the biomechanical properties (viscosity, engineering stress-strain, stress relaxation, *etc.*), apart from stiffness, to develop a realistic elastic deformable model.[145–148] Accurate models of clinically relevant tissues like, liver, gallbladder, kidney, and spleen can

help prevent potential damage to the tissue.

A comprehensive user study is also planned in the near future. In this study, we plan to recruit users to perform two tasks: palpation and ablation. Palpation is a key diagnostic aid for physicians to locate underlying tissue abnormalities. We will investigate the reliability and accuracy of the online stiffness estimation technique, in detecting the location of hidden stiff inclusions, using robot-assisted telemanipulation. This experiment will be performed on a mock anatomy. The ablation task will also be performed on a similar mock anatomy with an embedded rubber representing an artery. The user will be instructed to follow the artery with/without assistance and the effectiveness of the force-feedback and guidance will be investigated. Note that the robot will not be using any actual form of cautery/laser ablation.

Further, for a more interactive user-interface, we have an initial implementation of masters-as-mice. The goal is to develop a haptic engine that would convert the existing Qt based graphical user interface (GUI) to a haptic GUI. This will allow the developers to overlay a haptic plane with a task-specific interactive Qt GUI on the master console view. Our initial implementation is detailed in Appendix D

Appendix A

Constrained Optimization

Constrained optimization used for motion control of the slave and proxy slave robot used in 5.1 and 5.2 is detailed here. This chapter will provide an overview of the basic optimization framework as discussed in [149, 150], followed by a few examples of constrained motion. A series of linear equations is constructed that represent constraints, and a least squares solver is used to determine the "best" motion of the robot.

Objectives and Constraints

An objective is a term that the optimizer will try to minimize by its choice of a solution. Given multiple objectives, a solution will be found that best minimizes all of the corresponding expressions simultaneously. A constraint is a condition of an optimization problem that the solution must satisfy. Constraints can be of two types:

equality and inequality constraints.

- **Objective :** Objectives are of the form

$$\min_X \|AX - B\| \tag{A.1}$$

  where A is the objective matrix and B is the objective vector.

- **Equality Constraint :** Equality constraints are usually of the form

$$EX = F \tag{A.2}$$

  where E is the equality matrix and F is the equality vector.

- **Inequality Constraint :** Inequality constraints are usually of the form

$$GX \geq H \tag{A.3}$$

  where G is the inequality matrix and H is the inequality vector.

Example

A sample joint position control is detailed below, where the tip of the robot is always on or above a known plane.

Objective :

Minimize the distance between the current and the commanded cartesian position.

$$\min_{\delta q} \| J\delta q - (x_d - x_c) \|$$

where J is the Jacobian of the robot, x_d is desired/commanded cartesian position, x_c is the current cartesian position and δq is the incremental joint motion that is being optimized.

Plane Constraint

The goal here is to set up a virtual plane to constrain the motion of the tooltip to be on or above the plane. A plane passing through a point $\vec{x_p}$ with a normal $\vec{n}$ can be represented as

$$\hat{n} \cdot \vec{x_p} = d$$

The goal is to constrain the resulting robot position, after applying the δx increment, to be on or above the plane. So the constraint would be a

$$\begin{aligned} \hat{n} \cdot (\vec{x_c} + \delta x - \vec{x_p}) &\geq 0 \\ \hat{n} \cdot \delta x &\geq \hat{n} \cdot (\vec{x_p} - \vec{x_c}) \\ (\hat{n}^T J_p)\delta q &\geq (\hat{n}^T (\vec{x_p} - \vec{x_c})) \end{aligned}$$

where J_p corresponds to Jacobian for the position of the end-effector. This equation is of the form $GX \geq H$.

Joint Limit Constraints

For any kind of robot motion it is absolutely necessary to have limits on the joint motion for safety. Since the robot is controlled in joint space, we have joint limit constraints set up as follows,

$$l \leq q_c + \delta q \leq u$$

$$\underset{2n\times n}{\begin{bmatrix} I \\ -I \end{bmatrix}} * \underset{n\times 1}{\begin{bmatrix} \delta q \end{bmatrix}} \geq \underset{2n\times 1}{\begin{bmatrix} l - q_c \\ q_c - u \end{bmatrix}}$$

where l and u are the lower and upper joint position limits and q_c is the current joint state.

Appendix B

Impedance Virtual Fixture Example

A step-by-step example of a simple impedance virtual fixture/compliance wrench estimation is demonstrated using the CSA interface. We assume the reader has a fair knowledge of ROS subscribers and publishers which can be found in [151, 152].

Example

Let's assume we want to constrain the robot tip (p_r) to stay on a plane ($\mathcal{P}$) defined by a normal vector ($\hat{n}$) and origin (O) of the plane as shown in Figure B.1. For simplicity, we assume the plane $\mathcal{P}$ is in the robot frame and thus the compliance frame $F_c = [I, \vec{0}]$.

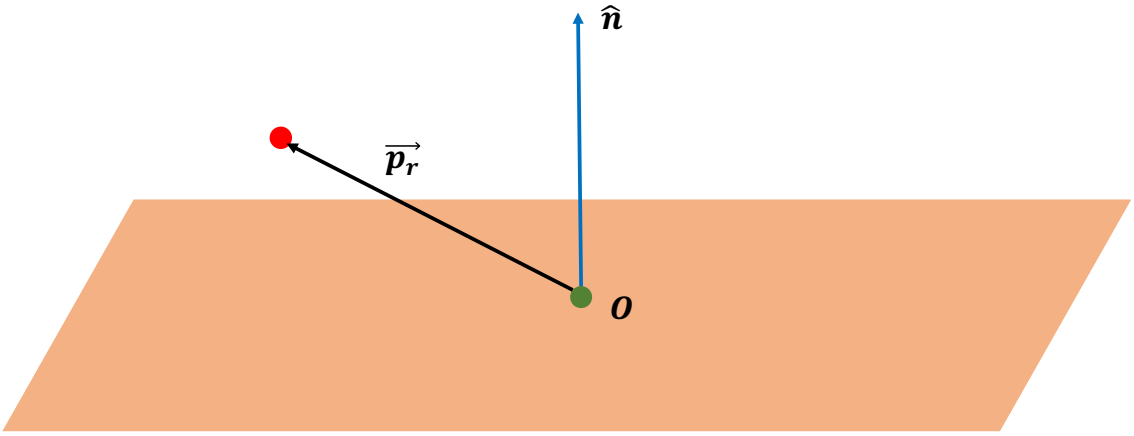


Figure B.1: Example Plane, where green dot represents point on the plane and red dot represents point above or below the plane

Compute

Only activate the fixture when the tip is within some threshold distance $\epsilon_d \approx 2mm$ from the plane, otherwise, a huge amount of force would be computed if the tip is far away. Once the robot tip $\vec{p_r}$ is within ϵ_d distance from $\mathcal{P}$, determine if p_r is below or above the plane. Thus, the position error $\vec{\epsilon}$ in Algorithm 5.1 is computed as follows:

$$\vec{\epsilon} = \vec{p_r} - \vec{p_c}$$

where p_c is the closest point on the plane, i.e, projection of $\vec{p_r}$ on $\mathcal{P}$. (Figure B.2)

Sample stiffness gains to calculate the compliance force for this fixture are as follows:

- Positive Stiffness Gain ($\vec{k}^+$) : vct3(0,0,-1000)
- Negative Stiffness Gain ($\vec{k}^-$) : vct3(0,0,-1000)

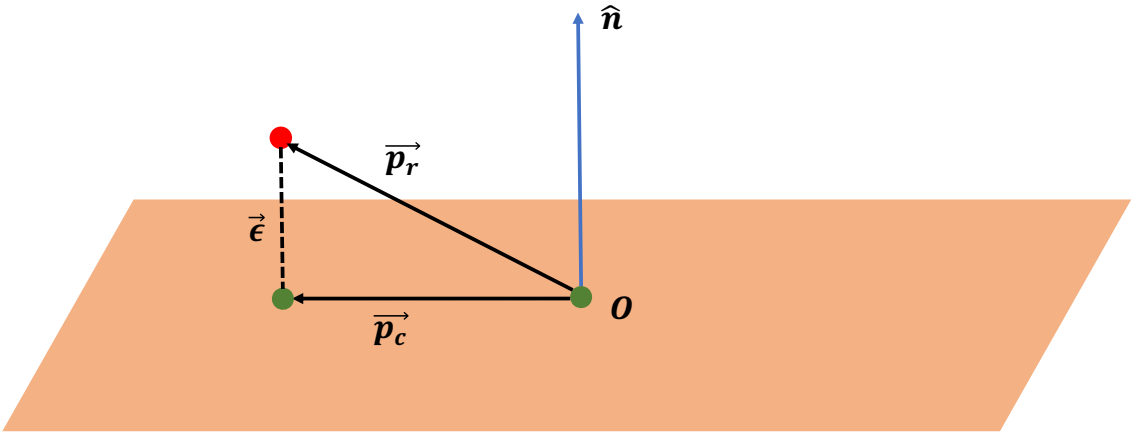


Figure B.2: Projection $\vec{p}_c$ of robot tip $\vec{p}_r$ onto the plane

Similarly, tip velocity $\dot{p}_r$ can be computed using the robot kinematics and necessary damping gains can be added

- Positive Damping Gain ($\vec{b}^+$) : vct3(-10, -10, -10)
- Negative Damping Gain ($\vec{b}^-$) : vct3(-10, -10, -10)

Bias force $\vec{g}^+, \vec{g}^-$ can also be added if we want the tip to slowly move towards the plane when it is far away.

These gain parameters need to be set once during the start of the program and internally CSA will turn on and off the impedance fixture based on the robot tip position. Complete pseudocode for this fixture is detailed in Algorithm B.1. Necessary ROS topics are available for the user to set these gain parameters using python or C++ platform and the corresponding ROS message is shown in Figure B.3.

Algorithm B.1 VF Example

Input: $\vec{p}_r$, $\dot{p}_r$

Initialize `VFGains` ▷ Can be hard coded or set externally

$[\vec{f}, \vec{\tau}]$ = Compute Compliance Wrench ▷ Algorithm 5.1

if $(acos(\vec{p}_r.\hat{n}) \leq 0)$ OR $(\epsilon_d \leq 2)$ **then**

Enable Fixture ▷ If the tip is close to the plane or below the plane

else

Disable Fixture

$[\vec{f}, \vec{\tau}] = \vec{0}$

end if

Send $[\vec{f}, \vec{\tau}]$ to Master

```
#prmCartesianImpedanceGains.msg
Header header

#VF position and orientation
geometry_msgs/Quaternion  ForceOrientation
geometry_msgs/Vector3     ForcePosition
geometry_msgs/Quaternion  TorqueOrientation

#Force gains
geometry_msgs/Vector3     PosStiffPos
geometry_msgs/Vector3     PosStiffNeg
geometry_msgs/Vector3     PosDampingPos
geometry_msgs/Vector3     PosDampingNeg
geometry_msgs/Vector3     ForceBiasPos
geometry_msgs/Vector3     ForceBiasNeg

#Torque gains
geometry_msgs/Vector3     OriStiffPos
geometry_msgs/Vector3     OriStiffNeg
geometry_msgs/Vector3     OriDampingPos
geometry_msgs/Vector3     OriDampingNeg
geometry_msgs/Vector3     TorqueBiasPos
geometry_msgs/Vector3     TorqueBiasNeg
```

Figure B.3: Impedance gains ROS Message

Appendix C

Cross-Platform Socket Based Communication

As mentioned earlier, the CSA framework is independent of the robotic platform. Any slave device with position control and any master device with torque control is supported. However, not all robotic platforms use cisst libraries. Some robotic platforms are developed on windows using the MATLAB Simulink® Real-Time™ environment and some are built using ROS. We have developed a socket based communication pipeline using a custom User Datagram Protocol (UDP) packet for network communication. This is a prototype version and is capable of communicating with a slave over the network. There are two components developed for this purpose; one on the server side (*SocketServer*) to get the information from the slave robot and send it over UDP and the other on the client side (*SocketClient*) which retrieves information

over UDP and sends it to the *Teleop* component. Figure C.1 shows the component connections of this setup. Note that the *SocketServer* component is only needed when the Slave robot supports cisst libraries and is on a different network, otherwise, the slave robotic platform needs to build a network interface to send/receive the custom UDP packet information. Complete packet information is provided in Figures C.2, C.3 and C.4.

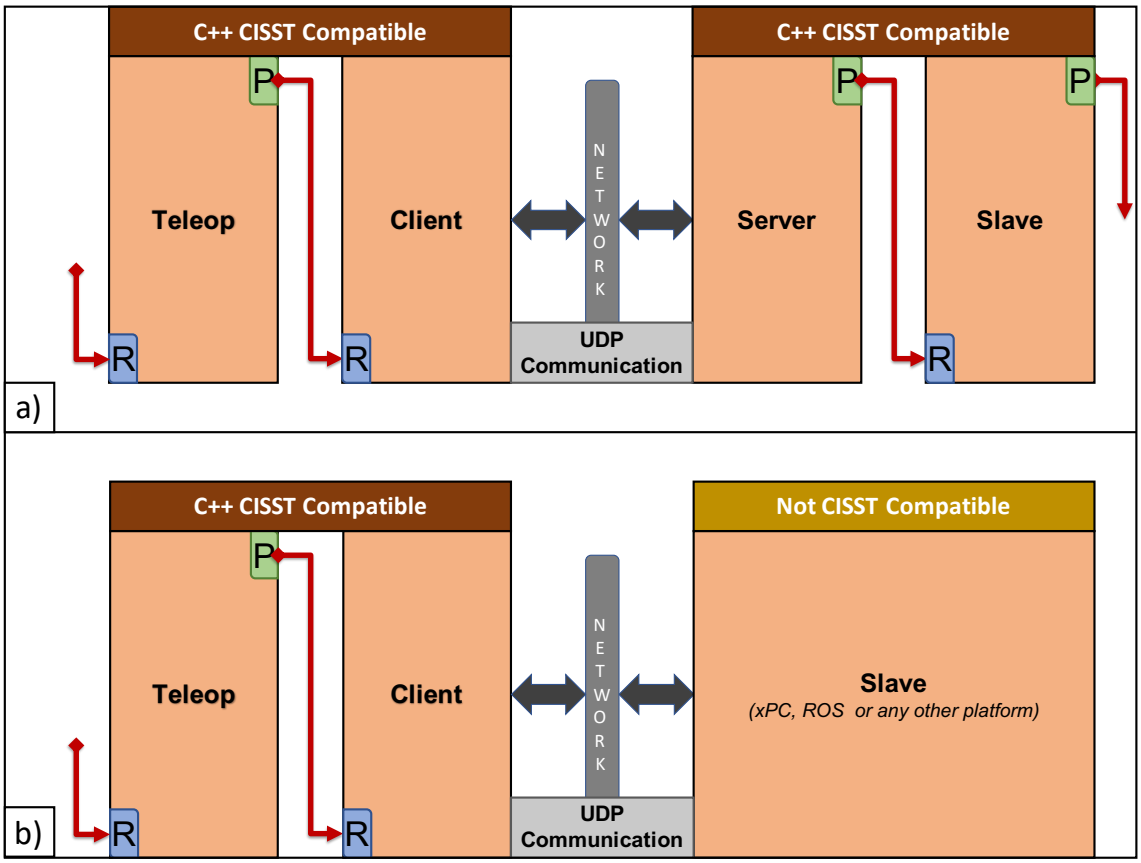


Figure C.1: UDP Slave configuration a) Cisst compatible Slave; b) Other Slave devices.

Value	StateType	Description
0	SCK_UNINITIALIZED	Used to turn off the system (Current/Desired)
1	SCK_HOMING	Used to indicate the arm is currently homing (Current)
2	SCK_HOMED	Used to Home/Power the arm (Current/Desired)
3	SCK_CART_POS	Used to set direct cartesian position (Current/Desired)
4	SCK_CART_TRAJ	Used to set trajectory cartesian goal (Current/Desired)
5	SCK_JNT_POS	Used to set direct joint position (Current/Desired)
6	SCK_JNT_TRAJ	Used to set trajectory joint goal (Current/Desired)

Table C.1: ***socketMessages*** : Enum values for robot states for socket based connection. Can be used to either report the current state or set a desired state.

```
socketHeader {
    //Version number. 10000 stands for 1.00.00
    int Version;

    //Message id counter. Increment by 1 and the first packet number
   starts with 0
    unsigned int Id;

    // Message size in bytes, including the header
    int Size;3

    // Local timestamp in seconds
    double Timestamp;

    // Last Message id received
    unsigned int LastId;

    // Timestamp in seconds of the last message received
    double LastTimestamp;
}
```

Figure C.2: Header for the socket state and command message

```
socketState {
    socketHeader Header;
    socketMessages::StateType RobotControlState;
    vctFrm3 CurrentPose;
    double CurrentJaw;
}
```

Figure C.3: socketState: Message used to report current state information of the arm

```
socketCommand {
    socketHeader Header;
    socketMessages::StateType RobotControlState;
    vctFrm3 GoalPose;
    double GoalJaw;
}
```

Figure C.4: socketCommand: Message used to send motion commands to the arm

Appendix D

Masters-as-Mice

Masters-as-mice is an interactive mode, where the user uses the master handle to control a 3D cursor. In this case, a haptic interactive plane is rendered on the console view and the developers can provide a task-specific user interface. The users would be able to interact with this interface using the system cursor, and haptic feedback would be provided upon interaction. There are three important tasks involved in developing the masters-as-mice interactive mode,

- **Use the master device to control the X11 cursor:**

  We make use of the X11 library to communicate with the X11 server and generate system level cursor motion events. We project the master handle's 3D motion onto a 2D virtual plane and generate a 2D X11 move/click event, as shown in Figure D.1. This 2D motion is then sent to the X11 server as system level cursor move/click events.

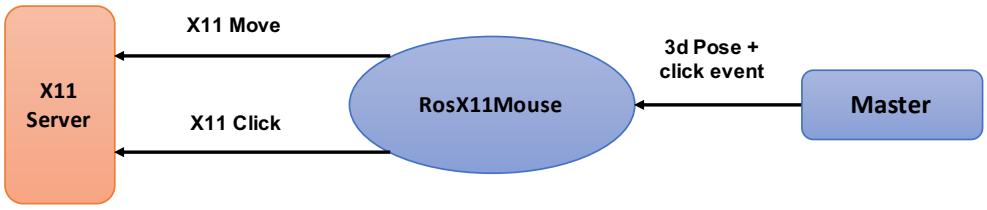


Figure D.1: X11 Communication

- **Convert Qt widget into a haptic widget:** A haptic engine runs in the background to constantly monitor the cursor location. The goal of this engine is to convert any Qt widgets under the cursor location to haptic widgets. This is done by adding some necessary Qt signal/slots for compliance wrench estimation based on the cursor location. Figure D.2 shows the workflow of the haptic engine. This would convert any existing Qt-based GUI to be used as a haptic interface for masters-as-mice, without any code modifications.

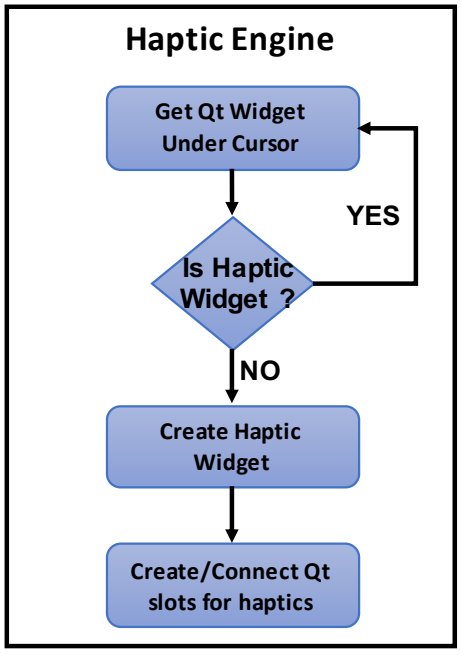


Figure D.2: Haptic engine

- **Define click/move/slider interactions with the 3D cursor:**

  Different types of Qt widgets need to have different interactions, thus, we need to define various compliance configurations based on widget type.

Figure D.3 shows the complete workflow to provide haptic feedback for the users upon interacting with the user interface, using the 3D cursor.

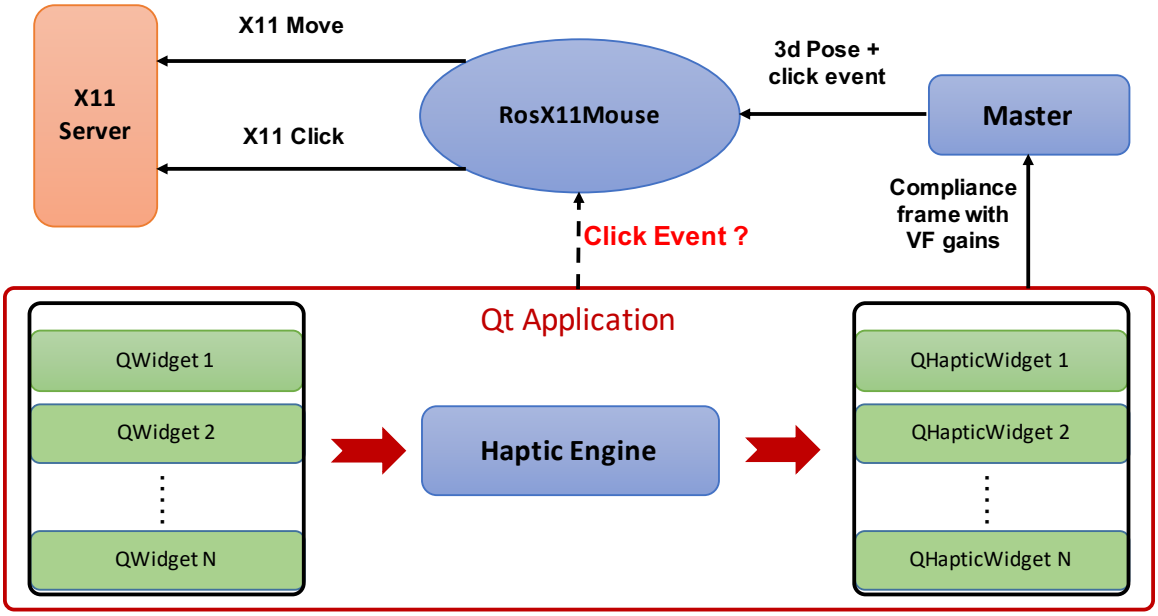


Figure D.3: Complete information flow in masters-as-mice mode

Acronyms

API Application Programming Interface. 35

CBCT Cone-Beam Computed Tomography. 154

CPD Coherent Point Drift. 44, 90, 92, 98, 145, 154, 156, 166

CT Computed Tomography. 148, 156

DOF Degree of Freedom. 7, 9, 110

dVRK da Vinci Research Kit. 5, 6, 8, 33, 36, 83, 119, 121, 132, 136, 143, 146, 149, 156–158, 165

ECM Endoscope Camera Manipulator. 158

FOV Field of view. 158

GMM Gaussian Mixture Model. 92

GP Gaussian Processes. 13, 43, 47, 48, 50, 53, 54, 57–62, 64, 66, 69, 70, 73, 77, 83–85, 90, 93, 94, 101, 103, 166, 168

Acronyms

GUI graphical user interface. 169, 183

IMLP Iterative Most Likely Point. 44, 90–94, 97, 166

IP Internet Protocol. 158

KD K-Dimensional. 91

LLC Low Level Controller. 27, 29–31, 120, 121, 125, 128, 131, 132, 135, 140

MIS Minimally Invasive Surgery. 1, 2, 4, 16, 19, 22, 39, 123, 164

MLC Mid Level Controller. 27, 29–31, 114–116, 120, 128, 131–134, 166, 167

MMT Model Mediated Teleoperation. 10, 11, 107–110, 129, 142, 166, 168

MTM Master Tool Manipulator. 6

NCC Normalized Cross-Correlation. 158

pCLE probe-based Confocal Laser Endomicroscopy. 157, 164

PD Principal Direction. 91

PSM Patient Side Manipulator. 6, 143, 146

RMS Root Mean Square. 54, 55, 88, 146

ROI region of interest. 152

Acronyms

ROS Robot Operating System. 6, 27, 36, 44, 117, 135, 158, 174, 176, 178

SAW Surgical Assistant Workstation. 33

STL Stereolithography. 94

TCP Transmission Control Protocol. 8, 158

UDP User Datagram Protocol. 178

Bibliography

[**1**] C. M. R. Marohn and C. E. J. Hanly, "Twenty-first century surgery using twenty-first century technology: surgical robotics," *Current surgery*, vol. 61, no. 5, pp. 466–473, 2004.

[**2**] Y. Wang, S. E. Butner, and A. Darzi, "The developing market for medical robotics," *Proceedings of the IEEE*, vol. 94, no. 9, pp. 1763–1771, 2006.

[**3**] R. A. Beasley, "Medical robots: current systems and research directions," *Journal of Robotics*, vol. 2012, 2012.

[**4**] M. Hoeckelmann, I. J. Rudas, P. Fiorini, F. Kirchner, and T. Haidegger, "Current capabilities and development potential in surgical robotics," *International Journal of Advanced Robotic Systems*, vol. 12, no. 5, p. 61, 2015.

[**5**] J. Troccaz, *Medical robotics*. John Wiley & Sons, 2013.

[**6**] R. H. Taylor, J. Funda, B. Eldridge, S. Gomory, K. Gruben, D. LaRose, M. Talamini, L. Kavoussi, and J. Anderson, "A telerobotic assistant for laparoscopic

surgery," *IEEE Engineering in Medicine and Biology Magazine*, vol. 14, no. 3, pp. 279–288, 1995.

[**7**] M. Gagner, E. Begin, R. Hurteau, and A. Pomp, "Robotic interactive laparoscopic cholecystectomy," *The Lancet*, vol. 343, no. 8897, pp. 596–597, 1994.

[**8**] B. Davies, R. Hibberd, M. Coptcoat, and J. Wickham, "A surgeon robot prostatectomya laboratory evaluation," *Journal of medical engineering & technology*, vol. 13, no. 6, pp. 273–277, 1989.

[**9**] S. Bann, M. Khan, J. Hernandez, Y. Munz, K. Moorthy, V. Datta, T. Rockall, and A. Darzi, "Robotics in surgery," *Journal of the American College of Surgeons*, vol. 196, no. 5, pp. 784–795, 2003.

[**10**] H. A. Paul, B. Mittlestadt, W. L. Bargar, B. Musits, R. H. Taylor, P. Kazanzides, J. Zuhars, B. Williamson, and W. Hanson, "A surgical robot for total hip replacement surgery," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on.* IEEE, 1992, pp. 606–611.

[**11**] R. M. Satava, "Robotic surgery: from past to future–a personal journey." 2003.

[**12**] K. Machida, Y. Toda, T. Iwata, M. Kawachi, and T. Nakamura, "Development of a graphic simulator augmented teleoperation system for space applications," in *Guidance, Navigation and Control Conference*, 1988, p. 4095.

[**13**] J. H. Park, "Supervisory control of robot manipulator for gross motions," Ph.D. dissertation, Massachusetts Institute of Technology, 1991.

[**14**] J. Funda, T. S. Lindsay, and R. P. Paul, "Teleprogramming: Toward delay-invariant remote manipulation," *Presence: Teleoperators & Virtual Environments*, vol. 1, no. 1, pp. 29–44, 1992.

[**15**] B. Hannaford, L. Wood, D. A. McAffee, and H. Zak, "Performance evaluation of a six-axis generalized force-reflecting teleoperator," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 620–633, 1991.

[**16**] M. J. Massimino and T. B. Sheridan, "Variable force and visual feedback effects on teleoperator man/machine performance," 1989.

[**17**] R. Ravindran and K. Doetsch, "Design aspects of the shuttle remote manipulator control," in *Guidance and Control Conference*, 1982, p. 1581.

[**18**] M. L. Turner, R. P. Findley, W. B. Griffin, M. R. Cutkosky, and D. H. Gomez, "Development and testing of a telemanipulation system with arm and hand motion," in *ASME IMECE Symp. on Haptic Interfaces*, 2000.

[**19**] T. B. Sheridan, "Space teleoperation through time delay: Review and prognosis," *IEEE Transactions on robotics and Automation*, vol. 9, no. 5, pp. 592–606, 1993.

[**20**] G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl, "Sensor-based space

robotics-rotex and its telerobotic features," *IEEE Transactions on robotics and automation*, vol. 9, no. 5, pp. 649–663, 1993.

[**21**] B. Hannaford, "Ground experiments toward space teleoperation with time delay," *Progress in Astronautics and Aeronautics*, vol. 161, pp. 87–87, 1994.

[**22**] A. J. Madhani, G. Niemeyer, and J. K. Salisbury, "The black falcon: a teleoperated surgical instrument for minimally invasive surgery," in *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No. 98CH36190)*, vol. 2. IEEE, 1998, pp. 936–944.

[**23**] A. Rovetta, R. Sala, X. Wen, and A. Togno, "Remote control in telerobotic surgery," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 26, no. 4, pp. 438–444, 1996.

[**24**] D. B. Camarillo, T. M. Krummel, and J. K. Salisbury Jr, "Robotic technology in surgery: past, present, and future," *The American Journal of Surgery*, vol. 188, no. 4, pp. 2–15, 2004.

[**25**] J. Binder and W. Kramer, "Robotically-assisted laparoscopic radical prostatectomy," *BJU international*, vol. 87, no. 4, pp. 408–410, 2001.

[**26**] J. Marescaux, J. Leroy, M. Gagner, F. Rubino, D. Mutter, M. Vix, S. E. Butner,

and M. K. Smith, "Transatlantic robot-assisted telesurgery," *Nature*, vol. 413, no. 6854, p. 379, 2001.

[**27**] G. S. Guthart and J. K. Salisbury, "The intuitive/sup tm/telesurgery system: overview and application," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 1. IEEE, 2000, pp. 618–621.

[**28**] J. Funda and R. Paul, "A symbolic teleoperator interface for time-delayed underwater robot manipulation," in *OCEANS'91. Ocean Technologies and Opportunities in the Pacific for the 90's. Proceedings.* IEEE, 1991, pp. 1526–1533.

[**29**] A. M. Madni, A. Freedy *et al.*, "Intelligent interface for remote supervision and control of underwater manipulation," in *Proceedings of the First Annual International Robot Conference, Long Beach, CA, June.* Institute of Electrical and Electronics Engineers, 1983, pp. 149–155.

[**30**] D. Yoerger, J. Newman, and J.-J. Slotine, "Supervisory control system for the jason rov," *IEEE Journal of Oceanic Engineering*, vol. 11, no. 3, pp. 392–400, 1986.

[**31**] D. Yoerger and J.-J. Slotine, "Supervisory control architecture for underwater teleoperation," in *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4. IEEE, 1987, pp. 2068–2073.

[**32**] S.-G. Hong, J.-J. Lee, and S. Kim, "Generating artificial force for feedback control of teleoperated mobile robots," in *Intelligent Robots and Systems, 1999. IROS'99. Proceedings. 1999 IEEE/RSJ International Conference on*, vol. 3. IEEE, 1999, pp. 1721–1726.

[**33**] K. Kawabata, T. Ishikawa, H. Asama, and I. Endo, "Mobile robot teleoperation using local storage," in *Control Applications, 1999. Proceedings of the 1999 IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 1141–1145.

[**34**] T. Makiishi and H. Noborio, "Sensor-based path-planning of multiple mobile robots to overcome large transmission delays in teleoperation," in *Systems, Man, and Cybernetics, 1999. IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on*, vol. 4. IEEE, 1999, pp. 656–661.

[**35**] K. Schilling and H. Roth, "Control interfaces for teleoperated mobile robots," in *Emerging Technologies and Factory Automation, 1999. Proceedings. ETFA'99. 1999 7th IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 1399–1403.

[**36**] N. Diolaiti and C. Melchiorri, "Teleoperation of a mobile robot through haptic feedback," in *Haptic Virtual Environments and Their Applications, IEEE International Workshop 2002 HAVE.* IEEE, 2002, pp. 67–72.

[**37**] O. J. Rösch, K. Schilling, and H. Roth, "Haptic interfaces for the remote control

of mobile robots," *Control Engineering Practice*, vol. 10, no. 11, pp. 1309–1313, 2002.

[**38**] J. Lim, J. Ko, and J. Lee, "Internet-based teleoperation of a mobile robot with force-reflection," in *Control Applications, 2003. CCA 2003. Proceedings of 2003 IEEE Conference on*, vol. 1. IEEE, 2003, pp. 680–685.

[**39**] T. H. Massie, J. K. Salisbury *et al.*, "The phantom haptic interface: A device for probing virtual objects," in *Proceedings of the ASME winter annual meeting, symposium on haptic interfaces for virtual environment and teleoperator systems*, vol. 55, no. 1. Citeseer, 1994, pp. 295–300.

[**40**] J. Marescaux, J. Leroy, F. Rubino, M. Smith, M. Vix, M. Simone, and D. Mutter, "Transcontinental robot-assisted remote telesurgery: feasibility and potential applications," *Annals of surgery*, vol. 235, no. 4, p. 487, 2002.

[**41**] J. Marescaux and F. Rubino, "The zeus robotic system: experimental and clinical applications," *Surgical Clinics*, vol. 83, no. 6, pp. 1305–1315, 2003.

[**42**] R. Aracil, M. Buss, S. Cobos, M. Ferre, S. Hirche, M. Kuschel, and A. Peer, "The human role in telerobotics," in *Advances in Telerobotics*. Springer, 2007, pp. 11–24.

[**43**] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the $daVinci^{TM}$ surgical system,"

in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 6434–6439.

[**44**] Z. Chen, A. Deguet, R. H. Taylor, and P. Kazanzides, "Software architecture of the da vinci research kit," in *Robotic Computing (IRC), IEEE International Conference on*. IEEE, 2017, pp. 180–187.

[**45**] 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.

[**46**] "Universal robots," https://universal-robots.com, 2018, [Online; accessed 23-September-2018].

[**47**] "Kuka robots," https://www.kuka.com, 2018, [Online; accessed 29-October-2018].

[**48**] P. Mitra and G. Niemeyer, "Model-mediated telemanipulation," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 253–262, 2008.

[**49**] X. Li and P. Kazanzides, "Task frame estimation during model-based teleoperation for satellite servicing," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 2834–2839.

[**50**] A. Sotiras, C. Davatzikos, and N. Paragios, "Deformable medical image registration: A survey," *IEEE Transactions on Medical Imaging*, vol. 32, no. 7, pp. 1153–1190, 2013.

[**51**] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2262–2275, Dec 2010.

[**52**] S. Billings and R. Taylor, "Iterative most likely oriented point registration," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2014, pp. 178–185.

[**53**] I. Wanninayake, L. Seneviratne, and K. Althoefer, "Estimation of tissue stiffness using a prototype of air-float stiffness probe," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 1426–1431.

[**54**] D. Uribe, R. Stroop, T. Hemsel, and J. Wallaschek, "Development of a biomedical tissue differentiation system using piezoelectric actuators," in *Frequency Control Symposium, 2008 IEEE International*, May 2008, pp. 91–94.

[**55**] A. Sabatini, P. Dario, and M. Bergamasco, "Interpretation of mechanical properties of soft tissues from tactile measurements," in *Experimental Robotics I*, ser. Lecture Notes in Control and Information Sciences, V. Hayward and O. Khatib, Eds. Springer Berlin Heidelberg, 1990, vol. 139, pp. 452–462. [Online]. Available: http://dx.doi.org/10.1007/BFb0042534

[**56**] J. Dargahi, S. Najarian, V. Mirjalili, and B. Liu, "Modelling and testing of a sensor capable of determining the stiffness of biological tissues," *Electrical and Computer Engineering, Canadian Journal of*, vol. 32, no. 1, pp. 45–51, Winter 2007.

[**57**] H. Liu, D. P. Noonan, B. J. Challacombe, P. Dasgupta, L. D. Seneviratne, and K. Althoefer, "Rolling mechanical imaging for tissue abnormality localization during minimally invasive surgery," *Biomedical Engineering, IEEE Transactions on*, vol. 57, no. 2, pp. 404–414, 2010.

[**58**] A. Talasaz and R. V. Patel, "Integration of force reflection with tactile sensing for minimally invasive robotics-assisted tumor localization," *IEEE Transactions on Haptics*, vol. 6, no. 2, pp. 217–228, 2013.

[**59**] M. Mahvash, J. Gwilliam, R. Agarwal, B. Vagvolgyi, L.-M. Su, D. D. Yuh, and A. Okamura, "Force-feedback surgical teleoperator: Controller design and palpation experiments," in *Haptic interfaces for virtual environment and teleoperator systems, 2008. haptics 2008. symposium on*, March 2008, pp. 465–471.

[**60**] G. Guthart and J. Salisbury, J., "The $INTUITIVE^{TM}$ telesurgery system: overview and application," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 1, 2000, pp. 618–621 vol.1.

[**61**] K. Xu and N. Simaan, "Intrinsic wrench estimation and its performance index

for multisegment continuum robots," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 555–561, 2010.

[**62**] ——, "An investigation of the intrinsic force sensing capabilities of continuum robots," *IEEE Transactions on Robotics*, vol. 24, no. 3, pp. 576–587, 2008.

[**63**] A. Bajo and N. Simaan, "Hybrid Motion/Force Control of Multi-Backbone Continuum Robots," *The International Journal of Robotics Research*, vol. 28, no. 9, pp. 1–13, July 2015.

[**64**] T. Xia, S. Leonard, I. Kandaswamy, A. Blank, L. Whitcomb, and P. Kazanzides, "Model-based telerobotic control with virtual fixtures for satellite servicing tasks," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013.

[**65**] S. Sanan, S. Tully, A. Bajo, N. Simaan, and H. Choset, "Simultaneous compliance and registration estimation for robotic surgery," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[**66**] J. Totz, S. Thompson, D. Stoyanov, K. Gurusamy, B. Davidson, D. Hawkes, and M. Clarkson, "Fast semi-dense surface reconstruction from stereoscopic video in laparoscopic surgery," in *Information Processing in Computer-Assisted Interventions*, ser. Lecture Notes in Computer Science, D. Stoyanov, D. Collins, I. Sakuma, P. Abolmaesumi, and P. Jannin, Eds. Springer International Publishing, 2014, vol. 8498, pp. 206–215.

[**67**] A. Schoob, D. Kundrat, L. Kahrs, and T. Ortmaier, "Comparative study on surface reconstruction accuracy of stereo imaging devices for microsurgery," *International Journal of Computer Assisted Radiology and Surgery*, pp. 1–12, 2015. [Online]. Available: http://dx.doi.org/10.1007/s11548-015-1240-z

[**68**] M. Parchami, J. Cadeddu, and G.-L. Mariottini, "Endoscopic stereo reconstruction: A comparative study," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*, Aug 2014, pp. 2440–2443.

[**69**] C.-Y. Lin, W.-T. Hung, and P.-J. Hsieh, *Stiffness Estimation in Vision-Based Robotic Grasping Systems*. Cham: Springer International Publishing, 2016, pp. 279–288.

[**70**] E. Ayvali, R. A. Srivatsan, L. Wang, R. Roy, N. Simaan, and H. Choset, "Using Bayesian optimization to guide probing of a flexible environment for simultaneous registration and stiffness mapping," *The International Conference on Robotics and Automation (ICRA)*, 2016.

[**71**] S. Caccamo, P. Gler, H. Kjellstrm, and D. Kragic, "Active perception and modeling of deformable surfaces using gaussian processes and position-based dynamics," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, Nov 2016, pp. 530–537.

[**72**] M. Hayashibe, N. Suzuki, and Y. Nakamura, "Laser-scan endoscope system for

intraoperative geometry acquisition and surgical robot safety management," *Medical Image Analysis*, vol. 10, no. 4, pp. 509–519, 2006.

[**73**] A. Garg, S. Sen, R. Kapadia, Y. Jen, S. McKinley, L. Miller, and K. Goldberg, "Tumor localization using automated palpation with gaussian process adaptive sampling," in *2016 IEEE International Conference on Automation Science and Engineering (CASE)*, Aug 2016, pp. 194–200.

[**74**] R. A. Srivatsan, E. Ayvali, L. Wang, R. Roy, N. Simaan, and H. Choset, "Complementary model update: A method for simultaneous registration and stiffness mapping in flexible environments," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 924–930.

[**75**] R. Inokuchi, H. Sato, Y. Aoki, and N. Yahagi, "Reminder of important clinical lesson: Bilateral common carotid artery dissection," *BMJ case reports*, vol. 2012, 2012.

[**76**] M. Jordan, S. Razvi, and M. Worthington, "Mycotic hepatic artery aneurysm complicating staphylococcus aureus endocarditis: successful diagnosis and treatment." *Clinical infectious diseases: an official publication of the Infectious Diseases Society of America*, vol. 39, no. 5, pp. 756–757, 2004.

[**77**] N. J. White, A. T. Sorkin, G. Konopka, and T. O. McKinley, "Surgical technique: static intramedullary nailing of the femur and tibia without intraop-

erative fluoroscopy," *Clinical Orthopaedics and Related Research®*, vol. 469, no. 12, pp. 3469–3476, 2011.

[**78**] G. W. Randolph, J. B. Kobler, and J. Wilkins, "Recurrent laryngeal nerve identification and assessment during thyroid surgery: laryngeal palpation," *World journal of surgery*, vol. 28, no. 8, pp. 755–760, 2004.

[**79**] S. P. Sterrett, T. Laurila, G. Bandi, and D. F. Jarrard, "Identification and preservation of accessory pudendal vessels during robot-assisted laparoscopic radical retropubic prostatectomy," *Journal of robotic surgery*, vol. 2, no. 1, pp. 31–34, 2008.

[**80**] D. Parmeggiani, G. Cimmino, D. Cerbone, N. Avenia, R. Ruggero, A. Gubitosi, G. Docimo, S. Mordente, C. Misso, and U. Parmeggiani, "Biliary tract injuries during laparoscopic cholecystectomy: three case reports and literature review," *Il Giornale di chirurgia*, vol. 31, no. 1/2, pp. 16–19, 2010.

[**81**] M. E. Allaf, A. W. Partin, and H. B. Carter, "The importance of pelvic lymph node dissection in men with clinically localized prostate cancer," *Reviews in urology*, vol. 8, no. 3, p. 112, 2006.

[**82**] C. K. Ng, I. S. Gill, M. B. Patil, A. J. Hung, A. K. Berger, A. L. de Castro Abreu, M. Nakamoto, M. S. Eisenberg, O. Ukimura, D. Thangathurai *et al.*, "Anatomic renal artery branch microdissection to facilitate zero-ischemia partial nephrectomy," *European urology*, vol. 61, no. 1, pp. 67–74, 2012.

[**83**] W. Saliba, V. Y. Reddy, O. Wazni, J. E. Cummings, J. D. Burkhardt, M. Haissaguerre, J. Kautzner, P. Peichl, P. Neuzil, V. Schibgilla *et al.*, "Atrial fibrillation ablation using a robotic catheter remote control system: initial human experience and long-term follow-up results," *Journal of the American College of Cardiology*, vol. 51, no. 25, pp. 2407–2411, 2008.

[**84**] C. Brace, "Thermal tumor ablation in clinical use," *IEEE pulse*, vol. 2, no. 5, pp. 28–38, 2011.

[**85**] G. D. Dodd III, D. Napier, J. D. Schoolfield, and L. Hubbard, "Percutaneous radiofrequency ablation of hepatic tumors: postablation syndrome," *American Journal of Roentgenology*, vol. 185, no. 1, pp. 51–57, 2005.

[**86**] B. J. Wood, J. Abraham, J. L. Hvizda, H. R. Alexander, and T. Fojo, "Radiofrequency ablation of adrenal tumors and adrenocortical carcinoma metastases," *Cancer*, vol. 97, no. 3, pp. 554–560, 2003.

[**87**] L. Swanstrom, M. Whiteford, and Y. Khajanchee, "Developing essential tools to enable transgastric surgery," *Surgical endoscopy*, vol. 22, no. 3, pp. 600–604, 2008.

[**88**] I. Ismail, R. Zhang, K. Ringe, S. Fischer, and A. Haverich, "Retrosternal adhesiolysis through an anterior minithoracotomy: a novel approach facilitating complete median redo sternotomy with a patent internal thoracic artery graft,"

The Journal of thoracic and cardiovascular surgery, vol. 137, no. 4, pp. 1034–1035, 2009.

[**89**] H. Hou, Y. Chen, X. Chen, C. Hu, Z. Yang, J. Chen, and X. Kong, "Related factors associated with pelvic adhesion and its influence on fallopian tube recanalization in infertile patients," *Zhonghua fu chan ke za zhi*, vol. 47, no. 11, pp. 823–828, 2012.

[**90**] R. ten Broek, B. van den Beukel, and H. Van Goor, "Comparison of operative notes with real-time observation of adhesiolysis-related complications during surgery," *British Journal of Surgery*, vol. 100, no. 3, pp. 426–432, 2013.

[**91**] M. T. Gettman, M. L. Blute, G. K. Chow, R. Neururer, G. Bartsch, and R. Peschel, "Robotic-assisted laparoscopic partial nephrectomy: technique and initial clinical experience with davinci robotic system," *Urology*, vol. 64, no. 5, pp. 914–918, 2004.

[**92**] M. Patel and J. Porter, "Robotic retroperitoneal partial nephrectomy," *World journal of urology*, vol. 31, no. 6, pp. 1377–1382, 2013.

[**93**] A. Deguet, R. Kumar, R. Taylor, and P. Kazanzides, "The cisst libraries for computer assisted intervention systems," in *MICCAI Workshop*, 2008.

[**94**] M. Y. Jung, M. Balicki, A. Deguet, R. H. Taylor, and P. Kazanzides, "Lessons learned from the development of component based medical robot systems," *J.*

of Software Engineering for Robotics (JOSER), vol. 5, no. 2, pp. 25–41, Sep 2014.

[**95**] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, Kobe, Japan, 2009.

[**96**] S. D. Billings, E. M. Boctor, and R. H. Taylor, "Iterative most-likely point registration (imlp): a robust algorithm for computing optimal shape alignment," *PloS one*, vol. 10, no. 3, p. e0117688, 2015.

[**97**] C. E. Rasmussen, "Gaussian processes for machine learning," 2006.

[**98**] M. Ebden, "Gaussian processes for regression: A quick introduction," *The Website of Robotics Research Group in Department on Engineering Science, University of Oxford*, 2008.

[**99**] K. O. Arras, "An introduction to error propagation: Derivation, meaning and examples of cy= fx cx fx," Tech. Rep., 1998.

[**100**] P. Kazanzides, J. F. Zuhars, B. D. Mittelstadt, and R. H. Taylor, "Force sensing and control for a surgical robot." in *ICRA*, 1992, pp. 612–617.

[**101**] P. Boyle and M. Frean, "Dependent gaussian processes," in *In Advances in Neural Information Processing Systems 17*. MIT Press, 2005, pp. 217–224.

[**102**] A. Lagae and P. Dutré, "Compact, fast and robust grids for ray tracing," in *Computer Graphics Forum*, vol. 27, no. 4. Wiley Online Library, 2008, pp. 1235–1244.

[**103**] S. Pabst, A. Koch, and W. Straßer, "Fast and scalable cpu/gpu collision detection for rigid and deformable surfaces," in *Computer Graphics Forum*, vol. 29, no. 5. Wiley Online Library, 2010, pp. 1605–1612.

[**104**] P. Chalasani, L. Wang, R. Yasin, N. Simaan, and R. H. Taylor, "Preliminary evaluation of an online estimation method for organ geometry and tissue stiffness," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1816–1823, 2018.

[**105**] L. Wang, Z. Chen, P. Chalasani, R. M. Yasin, P. Kazanzides, R. H. Taylor, and N. Simaan, "Force-controlled exploration for updating virtual fixture geometry in model-mediated telemanipulation," *Journal of Mechanisms and Robotics*, vol. 9, no. 2, p. 021010, 2017.

[**106**] P. Chalasani, L. Wang, R. Roy, N. Simaan, R. H. Taylor, and M. Kobilarov, "Concurrent nonparametric estimation of organ geometry and tissue stiffness using continuous adaptive palpation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 4164–4171.

[**107**] N. Verma, S. Kpotufe, and S. Dasgupta, "Which spatial partition trees are

adaptive to intrinsic dimension?" in *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 2009, pp. 565–574.

[**108**] R. S. J. Estépar, A. Brun, and C.-F. Westin, "Robust generalized total least squares iterative closest point registration," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2004, pp. 234–241.

[**109**] R. Balachandran and J. M. Fitzpatrick, "Iterative solution for rigid-body point-based registration with anisotropic weighting," in *Medical Imaging 2009: Visualization, Image-Guided Procedures, and Modeling*, vol. 7261. International Society for Optics and Photonics, 2009, p. 72613D.

[**110**] "Matlab engine api for c++," https://www.mathworks.com/help/matlab/matlab_external/matlab-engine-api-for-c.html, 2018, [Online; accessed 23-September-2018].

[**111**] R. Y. Rubinstein and D. P. Kroese, *The cross-entropy method: a unified approach to combinatorial optimization*. Springer, 2004.

[**112**] M. Kobilarov, "Cross-entropy motion planning," *International Journal of Robotics Research*, vol. 31, no. 7, pp. 855–871, 2012.

[**113**] E. Ayvali, A. Ansari, L. Wang, N. Simaan, and H. Choset, "Utility-guided

palpation for locating tissue abnormalities," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 864–871, April 2017.

[**114**] H. Salman, E. Ayvali, R. A. Srivatsan, Y. Ma, N. Zevallos, R. Yasin, L. Wang, N. Siman, and H. Choset, "Trajectory-optimized sensing for active search of tissue abnormalities in robotic surgery," *arXiv preprint arXiv:1711.07063*, 2017.

[**115**] E. Ayvali, H. Salman, and H. Choset, "Ergodic coverage in constrained environments using stochastic trajectory optimization," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 5204–5210.

[**116**] B. Hannaford, "A design framework for teleoperators with kinesthetic feedback," *IEEE transactions on Robotics and Automation*, vol. 5, no. 4, pp. 426–434, 1989.

[**117**] B. Willaert, J. Bohg, H. Van Brussel, and G. Niemeyer, "Towards multi-dof model mediated teleoperation: using vision to augment feedback," in *2012 IEEE Symposium on Haptic Audio-Visual Evironments and Games*, 2012, pp. 25–31.

[**118**] C. Passenberg, A. Peer, and M. Buss, "A survey of environment-, operator-, and task-adapted controllers for teleoperation systems," *Mechatronics*, vol. 20, no. 7, pp. 787–801, 2010.

[**119**] ——, "Model-mediated teleoperation for multi-operator multi-robot systems," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on.* IEEE, 2010, pp. 4263–4268.

[**120**] A. K. Bejczy, W. S. Kim, and S. C. Venema, "The phantom robot: predictive displays for teleoperation with time delay," in *Robotics and automation, 1990. proceedings., 1990 ieee international conference on.* IEEE, 1990, pp. 546–551.

[**121**] A. K. Bejczy and W. S. Kim, "Predictive displays and shared compliance control for time-delayed telemanipulation," in *Intelligent Robots and Systems' 90.'Towards a New Frontier of Applications', Proceedings. IROS'90. IEEE International Workshop on.* IEEE, 1990, pp. 407–412.

[**122**] W.-K. Yoon, T. Goshozono, H. Kawabe, M. Kinami, Y. Tsumaki, M. Uchiyama, M. Oda, and T. Doi, "Model-based space robot teleoperation of ets-vii manipulator," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 3, pp. 602–612, 2004.

[**123**] A. Alfi and M. Farrokhi, "Force reflecting bilateral control of master–slave systems in teleoperation," *Journal of Intelligent and Robotic Systems*, vol. 52, no. 2, pp. 209–232, 2008.

[**124**] X. Xu, B. Cizmeci, C. Schuwerk, and E. Steinbach, "Model-mediated teleoperation: toward stable and transparent teleoperation systems," *IEEE Access*, vol. 4, pp. 425–449, 2016.

[**125**] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 103, no. 2, pp. 126–133, 1981.

[**126**] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, February 1987.

[**127**] R. Featherstone, S. Sonck, and O. Khatib, *A general contact model for dynamically-decoupled force/motion control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 128–139. [Online]. Available: https://doi.org/10.1007/BFb0112956

[**128**] X. Guo, "Integrated active ultrasound systems for medical interventions," Ph.D. dissertation, 2015.

[**129**] A. Cheng, X. Guo, H. K. Zhang, H. J. Kang, R. Etienne-Cummings, and E. M. Boctor, "Active phantoms: a paradigm for ultrasound calibration using phantom feedback," *Journal of Medical Imaging*, vol. 4, no. 3, p. 035001, 2017.

[**130**] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Transactions on image processing*, vol. 10, no. 2, pp. 266–277, 2001.

[**131**] I. Mikic, S. Krucinski, and J. D. Thomas, "Segmentation and tracking in echocardiographic sequences: Active contours guided by optical flow esti-

mates," *IEEE transactions on medical imaging*, vol. 17, no. 2, pp. 274–284, 1998.

[**132**] M. Alemán-Flores, L. Álvarez, and V. Caselles, "Texture-oriented anisotropic filtering and geodesic active contours in breast tumor ultrasound segmentation," *Journal of Mathematical Imaging and Vision*, vol. 28, no. 1, pp. 81–97, 2007.

[**133**] B. Liu, H.-D. Cheng, J. Huang, J. Tian, X. Tang, and J. Liu, "Probability density difference-based active contour for ultrasound image segmentation," *Pattern Recognition*, vol. 43, no. 6, pp. 2028–2042, 2010.

[**134**] M. Hughes and G.-Z. Yang, "Line-scanning fiber bundle endomicroscopy with a virtual detector slit," *Biomedical optics express*, vol. 7, no. 6, pp. 2257–2268, 2016.

[**135**] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[**136**] T. Vercauteren, A. Meining, F. Lacombe, and A. Perchant, "Real time autonomous video image registration for endomicroscopy: fighting the compromises," *SPIE BIOS-Three-Dimensional and Multidimensional Microscopy: Image Acquisition and Processing XV*, vol. 6861, p. 68610C, 2008.

[**137**] R. J. Varghese, P. Berthet-Rayne, P. Giataganas, V. Vitiello, and G.-Z. Yang, "A framework for sensorless and autonomous probe-tissue contact management in robotic endomicroscopic scanning," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on.* IEEE, 2017, pp. 1738–1745.

[**138**] C. B. Duane, "Close-range camera calibration," *Photogramm. Eng*, vol. 37, no. 8, pp. 855–866, 1971.

[**139**] O. D. Faugeras, Q.-T. Luong, and S. J. Maybank, "Camera self-calibration: Theory and experiments," in *European conference on computer vision.* Springer, 1992, pp. 321–334.

[**140**] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on.* IEEE, 1997, pp. 1106–1112.

[**141**] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, 2000.

[**142**] L. Polok, V. Ila, M. Solony, P. Smrz, and P. Zemcik, "Incremental block cholesky factorization for nonlinear least squares in robotics." in *Robotics: Science and Systems*, 2013.

[**143**] L. Polok, M. Solony, V. Ila, P. Smrz, and P. Zemcik, "Efficient implementation

for block matrix operations for nonlinear least squares problems in robotic applications," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 2263–2269.

[**144**] V. Ila, L. Polok, M. Solony, and K. Istenic, "Fast incremental bundle adjustment with covariance recovery," in *3D Vision (3DV), 2017 International Conference on*. IEEE, 2017, pp. 175–184.

[**145**] M. Kauer, "Inverse finite element characterization of soft tissues with aspiration experiments," Ph.D. dissertation, ETH Zurich, 2001.

[**146**] I. Sakuma, Y. Nishimura, C. K. Chui, E. Kobayashi, H. Inada, X. Chen, and T. Hisada, "In vitro measurement of mechanical properties of liver tissue under compression and elongation using a new test piece holding method with surgical glue," in *Surgery Simulation and Soft Tissue Modeling*. Springer, 2003, pp. 284–292.

[**147**] A. Nava, E. Mazza, F. Kleinermann, N. J. Avis, J. McClure, and M. Bajka, "Evaluation of the mechanical properties of human liver and kidney through aspiration experiments," *Technology and Health Care*, vol. 12, no. 3, pp. 269–280, 2004.

[**148**] J. Rosen, J. D. Brown, S. De, M. Sinanan, and B. Hannaford, "Biomechanical properties of abdominal organs in vivo and postmortem under compression loads," *Journal of biomechanical engineering*, vol. 130, no. 2, p. 021020, 2008.

[**149**] J. Funda, R. H. Taylor, B. Eldridge, S. Gomory, and K. G. Gruben, "Constrained cartesian motion control for teleoperated surgical robots," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 3, pp. 453–465, 1996.

[**150**] A. Kapoor, M. Li, and R. H. Taylor, "Constrained control for surgical assistant robots." in *ICRA*, 2006, pp. 231–236.

[**151**] "Writing a simple publisher and subscriber (c++)," http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29, 2018, [Online; accessed 23-September-2018].

[**152**] "Writing a simple publisher and subscriber (python)," http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29, 2018, [Online; accessed 23-September-2018].

Vita

Preetham Chalasani received his Bachelor of Technology (B.Tech) degree in Information Technology from Indian Institute of Information Technology (IIIT), Allahabad in 2012. Later in the fall, he enrolled in Computer Science Masters program at Johns Hopkins university and in 2014 started his Ph.D. His research focussed on developing a real-time computational framework to provide situational awareness for the surgeons during telerobotic minimally invasive surgeries. The system is designed to simultaneously perceive the task environment and its operational constraints and uses the information for surgical guidance. He hopes this human-machine partnership will help surgeons in improving their understanding of the surgical scene and its correlation to pre-operative imaging-thereby increasing safety and improving surgical outcomes. During his Ph.D. he interned at Intuitive Surgical and starting the winter of 2018, Preetham will be joining the team at Intuitive Surgical.