TELE-OPERATION AND CONTROL OF ROBOTIC SURGICAL ASSISTANTS FOR SUTURING TASK

A Thesis

by

Begüm Sunal

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Computer Science

Özyeğin University
December 2021

TELE-OPERATION AND CONTROL OF ROBOTIC SURGICAL ASSISTANTS FOR SUTURING TASK

Advisor: Asst. Prof. Özkan Bebek

Co-advisor: Prof. Erhan Öztop

Approved by:

Asst. Prof. Özkan Bebek, Advisor
Department of Mechanical Engineering
Özyeğin University

Asst. Prof. Furkan Kıraç
Department of Computer Science
Özyeğin University

Assoc. Prof. Kemalettin Erbatur
Department of Mechatronics
Sabancı University

Date Approved: 29 December 2021

To my mom, dad and brother for their endless support, patience and trust.

To my loved one and friends who encouraged me through harsh times of this journey.

ABSTRACT

In a robot-assisted surgical systems, optimizations in tele-operation can lower the surgeon's physical and mental exhaustion and cognitive load, as well as the operation's completion time. As a result, it can considerably boost the chances of an operation's success. This thesis investigates several tele-operation methods to enhance the quality of the manipulation which will enable the surgeon to perform sensitive tele-operation tasks in narrow spaces quickly while reducing their mental load.

In the developed systems, it is ensured that the robot's end-effector configuration is automatically calculated among possible kinematic configurations in real-time. The presented methods have been tested in two highly redundant robotic surgical systems (DOF>=9) and benefits to the tele-operator have been demonstrated for each method by experiments that require dexterous manipulation. Experimental results show that using the developed methods in real-life operations will shorten the operation time and reduce the effort of the surgeon during the operation.

ÖZETÇE

Robotik destekli cerrahi sistemlerde, teleoperasyon üzerine yapılan iyileştirmeler, cerrahın fiziksel ve zihinsel yorgunluğunu, bilişsel yükünü ve operasyonun tamamlanma süresini azaltır. Sonuç olarak, bir operasyonun başarı şansı önemli ölçüde artırılır. Bu tez, manipülasyonun kalitesini artırarak, cerrahın zihinsel yüklerini azaltırken, dar alanlarda hassas teleoperasyon görevlerini hızlı bir şekilde gerçekleştirmesini sağlayacak çeşitli teleoperasyon yöntemleri önermektedir.

Geliştirilen sistemlerde robotun uç eleman konfigürasyonunun olası kinematik konfigürasyonlar arasından gerçek zamanlı olarak otomatik olarak hesaplanması sağlanmıştır. Sunulan yöntemler, yüksek derecede indirgenebilir iki robotik cerrahi sistemde (DOF>=9) test edilmiş ve her bir yöntem için sistemlerin teleoperatöre faydaları, hünerli manipülasyon gerektiren deneylerle gösterilmiştir. Deneysel sonuçlar, geliştirilen yöntemlerin gerçek hayattaki operasyonlarda kullanılmasının operasyon süresini kısaltacağını ve operasyon sırasında cerrahın eforunu azaltacağını göstermektedir.

ACKNOWLEDGEMENTS

The completion of this thesis would not have been possible without the expertises of my advisor Asst. Prof. Özkan Bebek and my co-advisor Prof. Dr. Erhan Öztop. I would also like to thank my committee members for taking time to read my thesis and listen to my defense.

I would like to extend my gratitude to Asst. Prof. Barkan Uğurlu for his valuable comments and feedbacks for my studies.

I would also like to thank my project members Negin Amirshirzad, Murat Özvin, Suavi Yıldırım and Oğulcan Çingiler, my lab partners Onur Ersoy and Barış Balcı, member of Biomechatronic Lab Erim Can Özçınar and member of the Interactive Intelligence Systems Lab Umut Çakan for their helps and comments.

Finally, I would like to thank my parents for their endless support through my journey. I could not have come this far without them on my side.

This research was done at Özyeğin University Robotics Laboratory and supported by the the Scientific and Technological Research Council of Turkey (TÜBİTAK), under grant number 119E036.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

In recent years, robotic systems in surgical environments have started to become more widely used and accepted as a better alternative to traditional surgery. Surgeries operated with robotic surgical assistants promise less pain, fewer complications and infections after the operation and faster healing process for the patients [1][2]. They also provide improved dexterity and manipulability to the surgeon inside small spaces [3].

1.1 Surgical Robotic Systems

There are several robotic system designs which are currently commercially available. These systems alleviate fatigue and reduce human errors while aiding the procedures that require high precision. The most popular example of commercially available surgical robotic system is the da Vinci® robotic system by Intuitive Surgical, shown on figure 1. The system consists of three components: the surgeon console, patient cart and vision cart. The surgeon console allows the surgeon to control the instruments with two haptic devices and pedals while viewing the surroundings in high-definition 3D. The Patient cart has four 6-DOF robotic arms holding the instruments and camera that the surgeon operates from the surgeon console. The vision cart allows the communication between the components while also supporting the 3D imaging.

Another commercially available surgical setup is Raven II by Applied Dexterity, shown on figure 2 which consists of two cable driven 7-DOF arms (6-DOF + grasp). The joints of the robotic arms are connected to the motors that are located at the base with cables and the joint angles are calculated using the encoders of the motors. The primary problem with the design of this system is that it does not consider stretches and slacks on the cables for joint angle calculation so the joint

Figure 1: Intuitive Surgical da Vinci® Surgical System (Image Courtesy of Intuitive Surgical, cropped, resized and licenced under CC BY-SA 4.0)

angles might be inaccurate if the cables are damaged.

Apart from the commercially available manipulators, many research teams all over the world have started to design their own generic or operation specific surgical systems. For example, figure 3 shows a surgical robot platform that was designed in Robotics Laboratory in Özyeğin University. The robotic arm has been developed to have a parallel linked structure and its control systems are specifically created for beating heart surgery [5][6].

1.2 Definitions of Concepts

Tele-operation is defined as manipulation of a system from a distance. Typically, a human operator controls a leader device from a distance in order to provide motion commands for a follower device which performs a task. For this thesis, a haptic device is used as a leader device which provides velocity inputs. These velocity inputs are either scaled or converted into positional inputs depending on the control scheme of the follower device, in this case a robotic arm.

Sometimes, the workspace of the leader device can be smaller than the workspace of the follower device. In that case, a *clutching* method should be added to the tele-operation system in case the limits of the leader device is reached. In

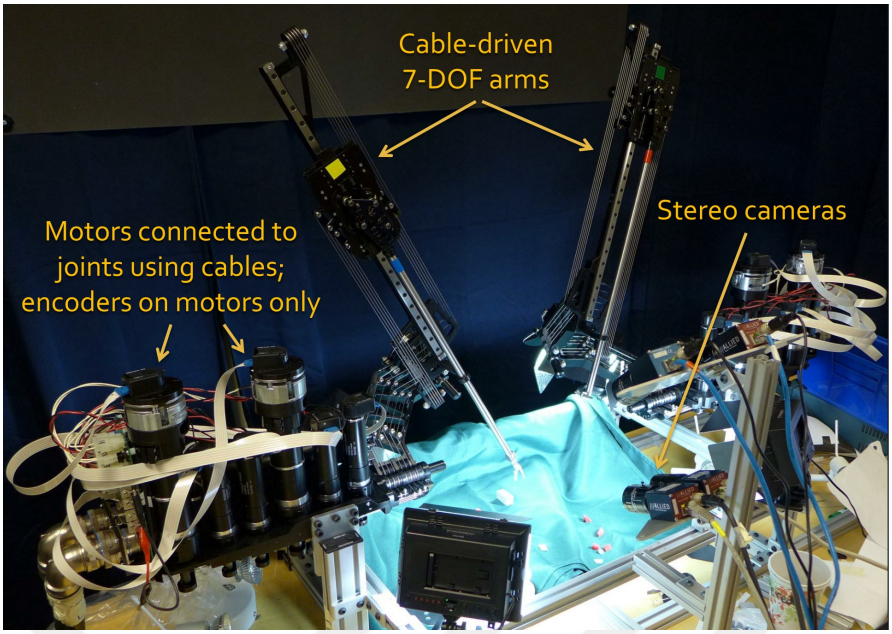


Figure 2: Applied Dexterity Raven-II Surgical Robot (Image Courtesy of Mahler et al. cropped, resized and retrieved from [4])

this thesis, pedals have been added to the system to enable clutching, inspired by daVinci® surgical robotic system. The tele-operator can send inputs for the follower device (i.e. the robotic arm) by pushing the pedal or reposition the leader device (i.e. the haptic device) by not pushing the pedal.

The robotic arms can be defined as *under-actuated*, *fully-actuated*, or *kinematically redundant* (also known as *over-actuated*) based on the number of joints they have. The manipulators that have same number of dimensions in cartesian and joint space (typically 6-DOF for 6 cartesian space dimensions) are called fully-actuated or actuated manipulators. The manipulators that have higher cartesian space dimensions than their number of joints (typically less than 6-DOF) are called under-actuated manipulators. Finally, the manipulators that have higher number of joints than cartesian space dimensions are called kinematically redundant or over-actuated manipulators.

The robotic systems that are used in this thesis are kinematically redundant manipulators with one of them having 9, and the other one having 10 joints, that operate in 6 dimensional cartesian space (3 translational, 3 rotational).

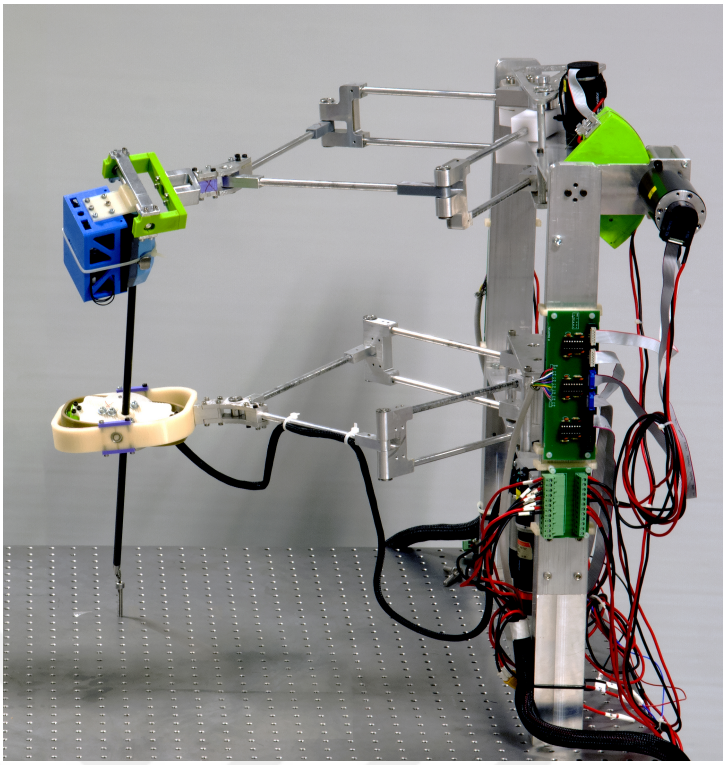


Figure 3: Ozyegin Surgical Robot Platform I designed for Minimally Invasive Robotic Surgery (MIRS) (Image Courtesy of Ersoy et al., retrieved from [5])

1.3 Objectives and Motivation

Even though the robot-assisted surgical systems are becoming better alternatives to the traditional surgical methods, the length of the operations may cause fatigue and increased cognitive load for the surgeons that are operating the robotic assistants. Thus, optimizations on the tele-operation of robotic systems is essential so that the surgeon can have a better focus on the procedure.

There are many ways to optimize the tele-operation of a robotic system. For a surgical robotic system, these optimizations should focus on reducing the effort and fatigue of the surgeon so that the success of the operation increases. In this thesis, many optimization methods that focuses on this motivation have been implemented. One of the methods aims to reduce the time of the procedure and the total number of clutches. Another method, aims to assist the surgeon to avoid physical limits of the joints and singular poses of the manipulator by sending force cues through the haptic device. Finally, the last method passively increases the

manipulability of the manipulator without interfering the tele-operator's intended trajectory.

Another way of reducing the fatigue of the surgeon would be to automate some of the surgical sub-tasks during the operation. Although, technological advancements are in the early stages on the automation of these surgical tasks, and a complete automous surgery is close to impossible with current technologies and knowledge, some of the sub-tasks of a surgery can be automated to reduce the burden of the surgeon tele-operator during a robot-assisted surgical operation.

Even though, the suturing task is one of the most challenging and time consuming of all the surgical sub-tasks [7], it would be a good surgical sub-task for automation due to its repetitive motions. Removing one of the hardest tasks of the surgeon helps reduce both mental and physical fatigue, thus would increase the success chance of other operations that cannot be automated. This thesis also elaborates autonomous control schemes of such a system. Specifically, it explains the concepts that are implemented that would make the autonomous control of the robotic system easier and more effective for automated suturing task.

1.4 Outline

The structure of the rest of the thesis is as follows: Chapter 2 provides some background information from the literature on the methods that are used in the thesis. Chapter 3 gives a generalized system overview, and explains the control schemes of all of the hardware that was used for various studies. Chapter 4 introduces different tele-operation optimizations and shares their results and verifications. Chapter 5 presents the tools that help with the autonomous execution of the suturing task. Finally, Chapter 6 concludes all of the concepts and presents the planned future work.

CHAPTER II

BACKGROUND

This chapter goes over the literature related to the studies that were conducted through the thesis. Section 2.1 inspects various tele-operation optimization techniques in literature. Section 2.2 explains the literature about autonomous manipulation of robotic systems for the suturing task.

2.1 Tele-operation Optimization

Throughout the years so many optimization techniques are developed for the tele-optimization of a manipulator. In this section, a portion of those techniques from the literature are introduced that are related to this thesis.

2.1.1 Adaptive Kinematics

A way of optimizing the tele-operation of a surgical system would be to allow the surgeon to do fine and precise movements or fast movements when necessary, so that the time of the operation, the tissue deformation and the number of clutching during the operation is reduced. Gras et. al. implemented an adaptive motion scaling by predicting the user's intended path using the combination of eye tracking and hand motion of the user [8].

A way of optimizing the tele-operation of a surgical system would be by taking advantage of the available redundancy. Redundancy in a robotic arm is explained as having extra freedom than needed in order to satisfy the position and orientation of the end effector. Thus, with a redundant robotic arm there are infinitely many solutions for inverse kinematics to reach every single point in the arm's workspace [9][10]. Ping et. al. proposes an optimal control method for redundant manipulators using the infinite number of solutions to their advantage. They have tested their algorithm on a 7-DOF manipulator, and they state that as the degree

of freedom increases, complexity increases exponentially. Thus, real-time control of higher degrees of freedom will be impossible [11].

Number of dimensions of kinematics solutions can be reduced by locking certain joints so that the manipulator could act like an actuated one rather than a redundant one. Tolani and Badler created a real-time controller by reducing the number of degrees of the human arm so that closed form solutions exist for the kinematics [12]. Chen et. al. and Zaplana and Basanez both introduced ways to select appropriate joint to be fixed so that the workspace of the manipulator is maximized [13][14]. All of the methods in this paragraph are tested on 7-DOF systems.

Xu et. al. has used recurrent neural networks (RNN) to reduce the errors that may occur because of pseudoinverse of Jacobian matrix [15]. However, the results of the method have been tested on a 6-DOF robot with only position tracking (i.e. three dimensional problem) which makes the manipulator redundant but simplifies the complexity regarding higher degrees of freedom manipulators with position and orientation tracking (i.e. six dimensional problem).

2.1.2 Haptic Guided Tele-operation

Haptic guided shared control is achieved by using force feedback of the haptic device to steer the user to the right position and orientation for the given task. Selvaggio et. al. proposed a haptic guidance system for optimizing the needle grasping in minimally invasive surgery [16]. They realized that joint limits and singularity configurations of the robot cause the surgeon to regrasp the needle multiple times and increase the fatigue. Thus, they have defined a cost function accounting for joint limits and another one for manipulability. By combining these two cost functions and applying gradient descent to find a local minimum, they acquired an optimal grasping parameter for the needle that will avoid joint limits and singularity poses [16].

Pocius et. al. realized in shared autonomy, human user feels like their authority no control is reduced and they are less willing to use the system [17].

Therefore, they have constructed a method that will optimize the teleoperation while retaining the control authority of the user. Kamale et. al. used two robots one being the master, one being the slave where the slave manipulator follows the movements of the master. Haptic Force Cues are generated for the master manipulator so that the user avoids singular positions, joint limits, and obstacles in the workspace [18]. Ghalamzan et. al. has used a similar approach for grasping an object task so that at the end of the grasping manipulability of the robot is maximized [19]. Force cues are sent to the haptic device using a task-oriented velocity manipulability cost function during the grasping of the object so that post-grasp pose of the robot has the maximum manipulability [19].

Selvaggio et. al. attempted to simplify the execution of the grasping task in a dual-arm system where one of the manipulators is holding a camera to track the object and moves completely autonomously and the other one is responsible for the grasping task that is teleoperated by the human user [20]. They have implemented several methods to simplify the grasping task for the user. First, orientation of the teleoperated manipulator is controlled autonomously when the user is only controlling the position of the robot. The orientation of the robot is always kept around the sphere of the object to grasp so that the user can always close into the object and immediately grasp. Second, the autonomous robot can track the teleoperated robot at all times and automatically avoid collisions so that the cognitive load of the human user is reduced. Third, haptic cues are introduced to the user where the robot is in near singular positions, joint limits, or collisions so that nonoptimal poses of the robot is avoided by haptic cues. All combinations of these methods (8 conditions) where they either use or do not use the specified methods are experimented on 15 subjects and it is found out that best results are yielded when all three methods are used. The subjects also found it easiest on the experiment when all three methods are used [20].

Several other studies can be found in literature that uses the haptic concept. Marchal-Crespo et al. researches the effect of haptic guidance while learning a

repetitive task [21]. Chinello et. al. compares the effects of haptic feedback and no haptic feedback during a manipulation task of a 6-DOF robotic arm for a wearable haptic device that they have designed [22]. Meli et. al. analyzes the affects of haptic feedback, generated and magnified from the forces on the operating table, on a tele-operation system robot-assisted surgical operations [23]. Bimbo et. al. uses a soft robotic hand that is attached to a 6-DOF robotic arm and a wearable arm-band to give force feedback for guiding the user towards the correct direction in a tele-operation experiment in cluttered environment [24].

2.1.3 Utilizing the Internal Motion

A kinematically redundant manipulator may reach a six dimensional pose in the cartesian space with an infinite number of joint configurations. That means the manipulator can move in the joint space without its end-effector pose. This property of redundancy is used for various studies in the literature. Shen et. al. and Dede et. al. proposed obstacle avoidance algorithms that use the internal motions of the 7-DOF manipulator to move away from the obstacles [25][26]. Zhu et. al. realizes that while utilizing the internal motions of the manipulator to avoid the obstacles, the manipulator sometimes cannot continue to execute their task motion. Therefore, they present a new approach where the manipulator continues its task motion while still avoiding the obstacles with its internal motion [27].

Dufour and Suleiman uses the internal motions for maximizing the manipulability of a robotic arm for offline trajectories [28]. Zhang et. al. introduces an online "self-motion planning" method to increase the manipulability of the manipulators formulated in the form of quadratic program. However, the only results of the method are from simulation environments with a 5-DOF and a 7-DOF planar robot, thus no physical experiments or experiments in higher cartesian spaces are conducted [29]. In another study, Zhang et. al. introduces the more refined version of the method: SM^3, "self-motion with manipulability maximization". This time the method is tested on a physical 6-DOF planar robot, however no experiments in higher cartesian spaces are conducted [30].

2.2 Suturing Task

Planning the autonomous suture beforehand would decrease the tissue trauma that may occur during the suturing process. Jackson and Çavuşoğlu suggested that at least 3 operations must be planned beforehand: entry and exit points of semi-circular needle, needle path, and the reaction needle gives in case of an unforeseen obstacle. With this goal, they have constructed an optimal path planning for surgical needle using the best practices of manual suturing [31]. Liu and Çavuşoğlu developed an algorithm that will select the correct orientation to grasp the needle and correct entry point so that number of regrasping therefore time of operation decreases [32]. Thananjeyan et. al. encountered that the robot is sometimes in near singularity configuration in the data collected from expert surgeons. Thus, they have proposed a fast running, dexterous solution to optimize the location and direction of the stitches that will avoid joint limits and singularities [33].

One of the earliest attempts in autonomous suturing is Endobot which has completely manual, shared control and completely autonomous modes [34]. Staub et. al. aimed to automate recurrent tasks in surgical setup in order to reduce the fatigue of the surgeon. With this goal, they have automated the tissue piercing where the surgeon points to the incision point with a laser pointer and the robotic assistant performs the stitching [35]. Iyer et. al. has tried to achieve a continuous needle insertion with a single arm, single camera system where the surgeon defines the entry and exit points. Although they achieved 85% success in terms of repeatability and accuracy, their main goal was to see if the automation of needle insertion was achievable with the single arm, single camera system rather than performing a full suture [36]. Sen et. al. combined a suture needle angular positioner (SNAP), a software for tracking the needle pose, and an optimization formulation of needle motion planning. With these concepts, they achieved an autonomous multi-throw suture system with 86% success rate and 30% faster than human operators [37].

Kam et. al. constructed a 3D path planning algorithm that will (i) detect

the NIR (Near Infrared) markers on deformable tissue (ii) generate a uniform and consistent suture placement based on locations of NIR markers, and (iii) update the suture plan after each completed stitch in order to create a semi-autonomous anastomosis robotic system. They have compared their results against manual laparoscopic suturing performed by an experienced surgeon and observed that their algorithm was more consistent in terms of bite depth and suture spacing that manual suturing [38].

Pedram et. al. implemented a path planning algorithm for suturing task that will minimize the tissue trauma and optimize the needle parameters such as shape and diameter to satisfy the recommended suturing guidelines [39]. Tian et. al. presented an autonomous surgical robotic system that can detect wounds and plan the path of the suturing needle using OCT (Optical Coherence Tomography) imaging. They report that the error is in sub-millimeter level between the planned and the executed entry and exit points of the needle [40].

CHAPTER III

SYSTEM OVERVIEW

This chapter explains a general overview of the systems and their control schemes that have been used for the rest of this thesis. In section 3.1, a generalized system overview is presented and the equipments that have been used are introduced. Section 3.2 explains various controllers that are implemented for the serial robotic arms. Section 3.3 presents controllers for the haptic devices. In section 3.4, the surgical tool control is described. Finally, section 3.5 introduces the simulation environment that have been used as an initial test setting.

3.1 Instrument Setup

To test and verify the accuracy of the systems offered, different experimental setups were constructed. Especially for the teleoperation studies, the experimental setups consist of some key hardware. These are a haptic device for receiving command inputs for the control system, a pedal that enables clutching, a serial manipulator as the main robotic arm, and a surgical tool as the end effector which is attached to the serial manipulator to perform surgical operations. In some cases, a secondary pedal was added to the system which limits the orientation input of the haptic device to command only the surgical tool rather than the whole system for easier control. For autonomous studies, only serial manipulators and surgical tools were used.

All controllers and software tools were modularly written and can be switched without affecting each other. For example, a joint impedance controller can be switched to a joint position controller without changing anything on the other modules, even though switching between a joint space controller and a cartesian space controller would require conversion on the input side. Similarly, a control system (i.e. conventional kinematics or an optimization method) can be switched

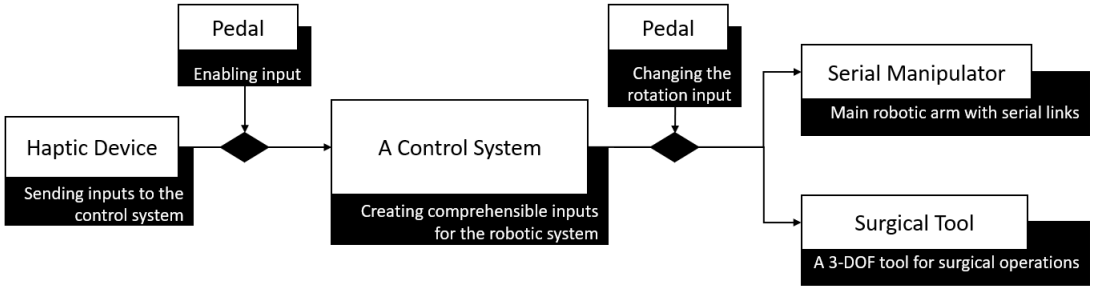


Figure 4: A generalized control flow for teleoperation

without changing the implemented controllers.

The communication between the modules was attained with Robot Operating System (ROS) through the nodes, topics, and services. The hardware is connected to one computer that runs on Ubuntu 18.04 Operating System with PREEMPT-RT real-time kernel patch. The serial manipulators were connected to the main computer with TCP/IP connections. Servo motor driver, the pedals, and the haptic devices were connected to the main computer with RS232 connections.

A rough generalization of an experimental setup for a teleoperation study is shown in Figure 4. The haptic device was used as an interaction tool between the teleoperator and the robotic arm. The user can give a target pose for the manipulator, and the haptic device can give force feedback which can notify the user about the robotic arm's status. Since the workspace of the haptic device is much smaller than the manipulator's workspace, a pedal was added to the system to attain clutching. The interaction between the haptic device and the manipulator is activated as long as the pedal is pushed so that the user can reposition the haptic device freely without sending any target positions to the robot.

In some cases, a finer movement of the surgical tool was needed to complete the operations (e.g. inserting the needle into a tissue during the suturing task). These types of movements can be hard to achieve by rotating the end effector frame of the whole robotic system with inverse kinematics calculations. Therefore, a second pedal was added to the system to switch the orientation control of the manipulator. If this secondary pedal was stepped on, the orientation input from

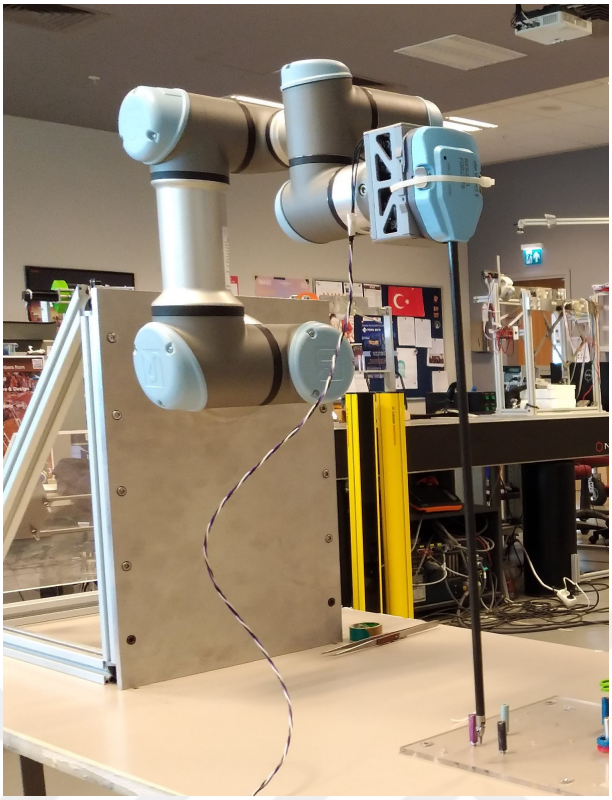


Figure 5: Universal Robots UR3 with a surgical tool attached to its end-effector

the haptic device was used to rotate only the 3-DOF wrist of the surgical tool, rather than the end effector of the entire system. During this manipulation, only the translation part of the inverse kinematics calculations of the whole system was calculated (i.e. in the x, y, and z directions).

The input poses from the haptic device were transferred to the serial manipulator and the surgical tool through a control system. These systems are further discussed in chapter 4. The control system generates comprehensible inputs for the robotic arm controllers which are explained in section 3.2, and the surgical tool controller which is explained in section 3.4.

The proposed systems have been tested on two different serial manipulators. In the earlier studies, a 6-DOF UR3 Robot (Universal Robots, Odense, Denmark) was used for experiments which is shown in Figure 5. Because UR3 Robot had some physical limitations in terms of control, later studies have been continued on

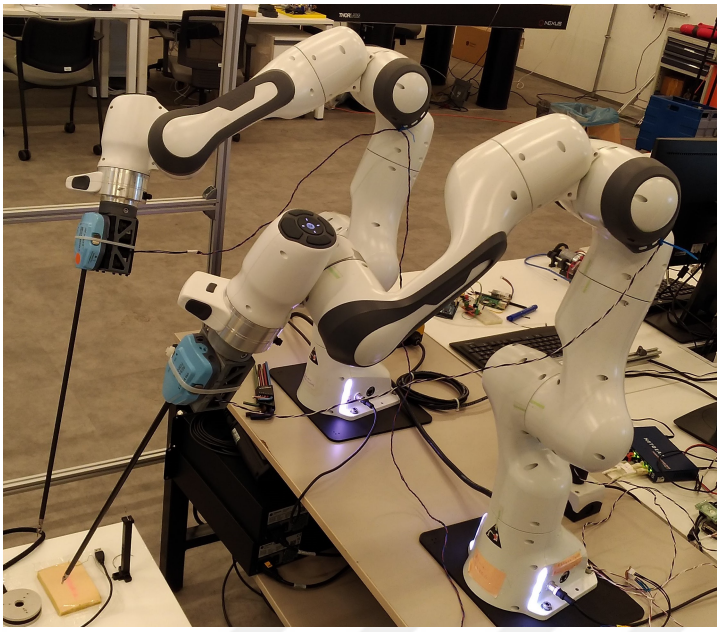


Figure 6: Dual Franka Emika Panda Manipulators with surgical tools attached on their end effectors

7-DOF Panda Robots (Franka Emika GmbH, München, Germany) shown in Figure 6. Different controllers for these robotic arms are discussed in Section 3.2. In both setups, a 3-DOF da Vinci surgical instrument (Intuitive Surgical, California, USA) has been attached to the end effector of the manipulator, namely either an 8 mm Mega Needle Driver or an 8 mm Resano Forceps. These surgical tools are driven by four XL-320 servo motors each (Robotis Co., Seoul, South Korea). The control of the surgical tools is explained further in Section 3.4.

Similarly, two different haptic devices were used in different setups. First, a 6-DOF omega.6 (Force Dimension, Nyon, Switzerland) haptic device with parallel joints have been used for some experiments that are shown in Figure 7. Second, a 6-DOF Touch (3D Systems, South Carolina, USA) haptic device with serial links was used that is shown in figure 8.

3.2 Robotic Arm Control Schemes

The controllers for both robotic arms were built as independent modules that take joint angles or Cartesian poses as input and bring the robot into its target configuration. In this way, the modularity of the overall system was established.

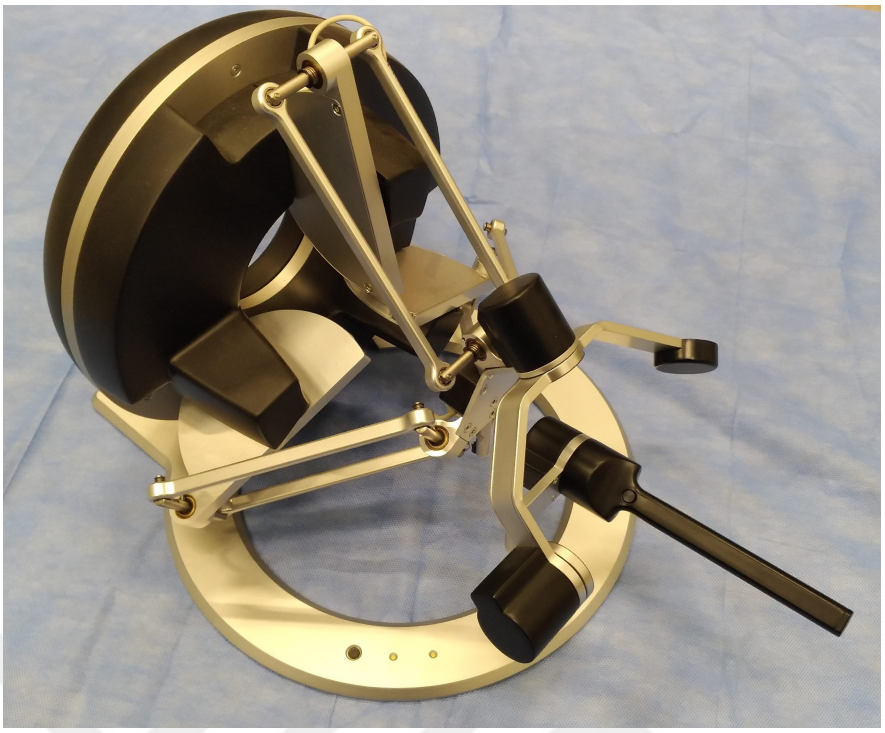


Figure 7: Force Dimension omega.6 haptic device

Figure 8: 3D Systems Touch haptic device

Therefore, the same controller can be used for different operations. In the cases where the inverse kinematics was externally calculated, the joint angles were provided as the input to the controller. In other cases, the cartesian poses were given to the controller as the input and the manipulator's internal kinematics were used to find the goal angles. For both robots (UR3 and Panda), position and velocity controllers were implemented. Since UR3 Robot does not support effort inputs, impedance controllers could only be implemented for the Panda robot.

To send commands to the UR3 Robot, a TCP/IP socket connection was created between the main computer and the robot. Then, target pose and/or angle commands were sent through this connection using UR Script Language. The maximum sampling frequency of 125 Hz was achieved with the UR3 Robot.

Panda robot controllers were written using $franka\_ros$ library, which is a library that integrates $libfranka$, i.e. open-source C++ interface of the Panda robot, to ROS (Robot Operating System) ecosystem. Panda robots can be controlled in real-time at 1 kHz using the ROS library.

3.2.1 UR3 Control Scheme

For the UR3 robot, a joint position control interface was implemented where the target joint angles were taken as the input and the robot was brought to said target joint angles. In most of the commands, UR3 internally generates a trajectory where it calculates every intermediate acceleration and velocity of the robotic arm between its current pose and the target pose. This introduces a vibration issue in a real-time scenario such as teleoperation. Since there was a constant stream of data provided from the haptic device or controller module to the robotic arm, UR3 generated a "safe trajectory" for itself in between every received target angle or pose. Therefore, the robot tried to speed up and then slow down at every time step. To avoid this start-stop motion, a special command (i.e. $servoj$) with a gain that works similarly to a proportional gain of the PID controller and a look ahead time that adjusts the velocity of the robotic arm was used. Thus, the robotic arm could be brought to the target joint angles without changing the velocity so often

when there was a continuous stream of input data.

3.2.2 Panda Control Schemes

Multiple control interfaces were implemented to be used in different scenarios for the Panda robot. For the control in cartesian space a velocity control interface, for the control in joint space a joint position control interface and a joint impedance control were implemented. Furthermore, the joint impedance control interface is extended for the dual Panda Robot system that was shown in figure 6.

The cartesian velocity control interface took target end-effector velocities as input and sent them to the robot after filtering with (1). Both for the safety of the robotic arm and the smoothness of the control, the input velocities were not directly fed to the system. Instead, (1) gradually increases or decreases the end-effector velocity of the manipulator up to the target velocities. In case there was no input to the system, the velocity was decreased using a constant decay rate until there was new input or the velocity reaches zero.

$$V_{filtered}(k+1) = \alpha V_{target}(k) + (1-\alpha)V_{filtered}(k) \tag{1}$$

where $V_{filtered}$ is the end-effector velocity, α is the linear filtering constant, V_{target} is the target velocity, k is the time index.

The joint position control interface takes the target joint angles as input and moves the robot to the target joint angles using a similar filtering method as the cartesian velocity interface. The controller interface gradually moves the joints to the target angles safely by using (2). Contrary to the cartesian velocity interface, no decay rate was used in the joint position interface since the joints stop moving when they reach their target angles.

$$q_{filtered}(k+1) = \alpha q_{target}(k) + (1-\alpha)q_{filtered}(k) \tag{2}$$

where $q_{filtered}$ is the joint velocity, α is the linear filtering constant, q_{target} is the target joint velocity, k is the time index.

The joint impedance control interface was the most widely used control scheme for the Panda robot across the studies that are presented in this thesis. Since the kinematics of the robotic system was manually solved (i.e. there is no need for the manipulator's internal kinematics solver), a joint space control scheme was needed rather than a cartesian one. Since control effort is much smoother than simple position control, impedance control interface was preferred over joint position interface.

Impedance control interface takes target joint angles as input, then calculates the difference between the target angles and current angles, represented with e (i.e. error) in (3). Then the difference of the error compared to the previous step was found by dividing the difference in errors with the period T as shown in (4). The efforts for each joint were found by predetermined stiffness, K, and damping, D, parameters in (5). Calculated efforts were saturated using (6). τ_{Jd} in (6) was the desired link-side joint torque sensor signal, which was received from Panda's internal model library, without the gravity compansation component.

$$e(k) = q_{target}(k) - q_{current}(k) \tag{3}$$

$$\dot{e}(k) = \frac{e_t(k) - e_{t-1}(k)}{T} \tag{4}$$

$$\tau_{calculated}(k) = Ke(k) + D\dot{e}(k) \tag{5}$$

$$\tau_{saturated} = \tau_{Jd} + \max(\min(\tau_{calculated} - \tau_{Jd}, \tau_{max}), -\tau_{max}) \tag{6}$$

Multi-arm systems, that is when two or more Panda manipulators are used, only control effort interfaces are supported. Therefore, either cartesian or joint space impedance control schemes can be used for such systems. For the dual Panda manipulator system, which is shown in Figure 6, the joint impedance control scheme explained above was extended for two robotic arms.

3.3 Haptic Device Control Schemes

In studies conducted throughout the thesis haptic devices were used as an interaction tool between the user and the robotic system. The user supplied the input for

the robotic system and the haptic device provided force feedback that informed the user about the manipulator's status. As explained in Section 3.1, a pedal was added to the system to enable the inputs from the haptic device and a second pedal was added for switching the orientation input of the haptic device from full system orientation to surgical tool's wrist orientation.

In order to scale the inputs from the haptic device to the robotic arm in a more straightforward fashion, the velocities of the haptic device in cartesian space were used instead of the position. Linear and angular velocity values of the haptic device were received from the Force Dimension SDK and published to the ROS network for omega.6 (shown in Figure 7).

For the Touch haptic device (shown in figure 8) linear velocities of the haptic device can be received from the device drivers, however, there is no support for the angular velocities. Therefore, angular velocities of the haptic device were estimated using the second-order one-sided backward difference method (7). Two time-step histories of the angular position of the haptic device (i.e. $X(k)$) were kept in memory and used for the calculation of the estimated angular velocities (i.e. $V(k)$).

$$\mathbf{V}(k) = \frac{3X(k) - 4X(k-1) + X(k-2)}{2T} \tag{7}$$

The velocities that were received from the haptic devices were scaled for the manipulator and used for either the velocity controllers or for iterating the inverse kinematics of the robotic system.

3.4 Surgical Tool Control Scheme

Throughout this thesis work, a 3-DOF servo-driven surgical tool (Intuitive Surgical, California, USA), that was attached to the serial manipulator, was used. This 3-DOF surgical tool was actuated by four Dynamixel servo motors (Robotis Co., South Korea) that are connected to the main computer with a RS232 connection.

The surgical tools that have been used were built with pulleys and cables in

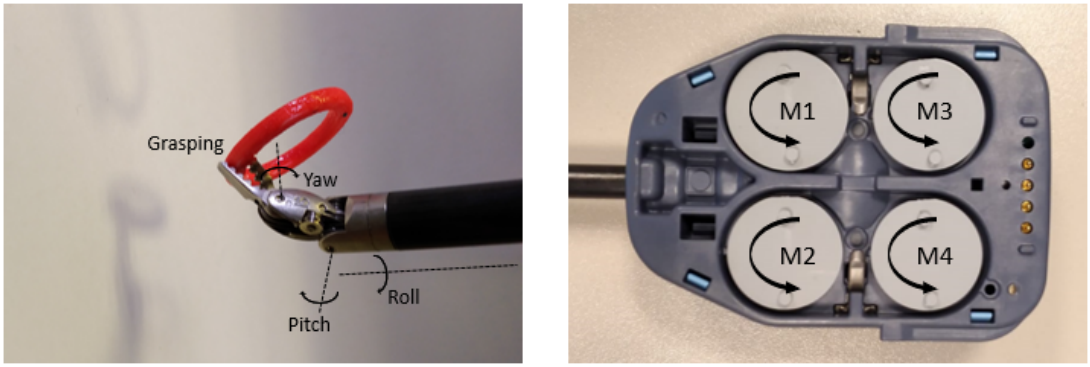


Figure 9: Left: wrist of the Endowrist needle driver which is capable of 3-DOF motion and grasping. Right: The four actuation input shafts of the surgical tool.

order to imitate the dexterity of a human wrist. Therefore, the three yaw, pitch, and roll motions of the tool were achieved with four rotations which introduced a decoupling problem to the system. Several motors must be rotated at different angles to get the desired pitch and yaw motions while keeping the other angles stationary. The motions at the wrist and the actuation shafts of the surgical tool are shown in Figure 9. On the left picture, yaw, pitch, roll, and grasping motions of the end effector are shown. On the right picture, the actuation shafts for these motions are shown. The roll motion of the end effector was achieved by rotating the third motor (shown with M3 in the figure). A pure yaw motion was achieved by combining motions from the first and the second motors (i.e. M1 and M2). A pure pitch motion is achieved by combining the motions of the first, the second, and the fourth motors (i.e. M1, M2, and M4).

The kinematics module sent all roll, pitch, and yaw motions at the same time to the servo controller at the same time. Thus, forming individual functions for each movement would not work. All motions must be calculated concurrently. Therefore, a transformation matrix was created in (8), which maps the input roll, pitch, and yaw motions to four motor angles. In this equation, k_r, $k_{p,i}$ and $k_{y,i}$ constants represents the rotation coefficients that were needed for each motion, θ_{Mi} represent the motor angles and $\theta_7, \theta_8, \theta_9$ are the desired wrist rotations sent by the kinematics module.

$$\begin{bmatrix} 0 & k_{p1} & k_{y1} & \theta_{M1_{offset}} \\ 0 & k_{p2} & k_{y2} & \theta_{M2_{offset}} \\ k_r & 0 & 0 & \theta_{M3_{offset}} \\ 0 & k_{p3} & 0 & \theta_{M4_{offset}} \end{bmatrix} \begin{bmatrix} \theta_7 \\ \theta_8 \\ \theta_9 \\ 1 \end{bmatrix} = \begin{bmatrix} \theta_{M1} \\ \theta_{M2} \\ \theta_{M3} \\ \theta_{M4} \end{bmatrix} \tag{8}$$

3.5 Simulation Environment

A simulation environment with the same dimensional and physical properties as the real-life system was created in Gazebo to use in the first trials of the studies. A screen caption of the simulation environment is shown in Figure 10. Panda robots are constructed using the technical drawings provided by the manufacturer. The surgical tool was constructed with boxes and cylinders which are attached to the end-effectors of the Panda robots. The robotic systems were elevated from the ground with a stand that had the same height as the table used to mount the robot arms in the real-life application. Also, an operation table was added in front of the arms. Dual-arm robotic system was used for both comparing different control systems next to each other and for the dual-arm studies.

The joints of the simulated robotic arms were controlled by joint position controllers. PID (proportional-integral-derivative) parameters were adjusted for each joint to obtain a more stable motion in the simulation.

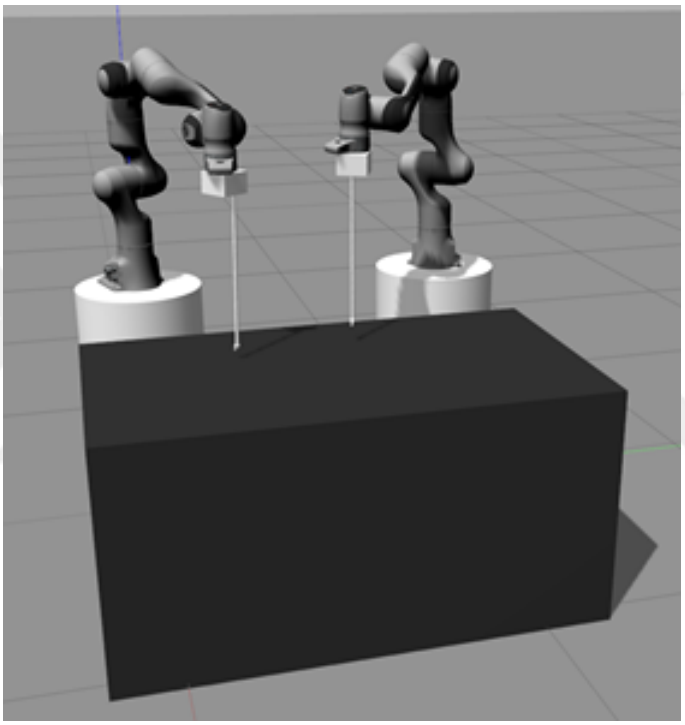


Figure 10: The simulation environment that is designed for initial experiments

CHAPTER IV

TELE-OPERATION SYSTEMS

This chapter explains the tele-operation systems that were implemented for the manipulators which was introduced in chapter 3. The section 4.1 introduces some essential insight on the topics that are referenced throughout the chapter. It also explains the process of modelling and solving the kinematics of the manipulators that are utilized. In section 4.2, a study where the velocity of the end-effector is scaled by locking certain joints of a redundant manipulator at certain times so that it behaves like an actuated one is proposed. Section 4.3 introduces a method to provide haptic force cues to the tele-operator in case of a non-optimal configuration of the manipulator. Section 4.4 proposes a method to maximize the manipulability index of the robotic arm by utilizing the internal motions that arise from the kinematic redundancy, without any motion on the end-effector pose. Finally, section 4.5 shows the experimental results and comparisons of the systems that are explained in sections 4.3 and 4.4.

4.1 Conventional Kinematics

This section of the thesis explains how teleoperation of the highly redundant robotic arms was achieved using conventional kinematic methods. In sub-section 4.1.1 first, the modeling of the two robotic systems, which consist of either a 6-DOF UR3 or 7-DOF Panda serial manipulator, with a 3-DOF surgical tool attached to the end-effector, is explained . Then, the Jacobian matrix calculations of these 9-DOF and 10-DOF systems are explained in sub-section 4.1.2. Finally, inverse kinematics calculations of said systems are discussed in sub-section 4.1.3.

4.1.1 Modelling the Robotic Systems

A robotic arm can be defined as a chain of links, that are a set of stationary bodies and their joints, that allow motion to their connected links. The kinematics of a robotic arm investigates its geometrical motion without considering any forces involved. The motion of a robotic arm can be achieved through the movements of its joints. These *joint space* motions can be converted to the *cartesian space* (also known as *task space*) by progressing through the kinematic chain of the robotic arm. The cartesian space mappings of the robotic arm can be found by setting the base of the manipulator as the origin and advancing through each link until the $end-effector$ of the robotic arm (i.e. tip of the last link).

Depending on how many joints a robotic arm has, it can be described as $under-actuated$, *actuated*, or *kinematically redundant*. A robotic arm with 6-DOF is considered an actuated manipulator since its 6 joints allow it to move in 6-dimensional cartesian space. A manipulator with less than 6-DOF is recognized as under-actuated since at least one of the motions in cartesian space mathematically cannot be computed. A manipulator with more than 6-DOF is defined as a kinematically redundant manipulator since any 6-dimensional movement in cartesian space can be reached in infinitely many joint configurations. The manipulators that have been used throughout the thesis are kinematically redundant robotic systems with one of them having 9-DOF and the other one having 10-DOF.

$$\vec{q} = \begin{bmatrix} q_1 \\ q_2 \\ \dots \\ q_m \end{bmatrix} \tag{9}$$

The joint space of a robotic arm with m joints can be defined with a column vector of size mx1 with m joint parameters as in (9). There are many ways to define the orientation of a robotic arm such as rotation matrices, quaternian

angles, euler angles etc. In this thesis, the cartesian space of the robotic arm's end-effector was defined with 3 translational positions (i.e. x, y and z), and 3 rotations defined using Euler angles (i.e. ϕ, θ, and ψ) as shown in (10).

$$\vec{X} = \begin{bmatrix} x \\ y \\ x \\ \phi \\ \theta \\ \psi \end{bmatrix} \tag{10}$$

Computing the end-effector position of a manipulator by using the joint parameters is called *forward kinematics* as shown in (11). On contrary, the computation of joint parameters that can bring the manipulator to a given end-effector pose in cartesian space is called *inverse kinematics* as shown in (12). The forward kinematics of the manipulators that are mentioned in this thesis were calculated using the Euclidean transformation method. The method considers the base of each link as a frame in 3 dimensions and creates a transformation matrix with (13) that defines the relative movement between each joint. The product of these homogeneous transformation matrices for each link results in the transformation of the end-effector from the origin which is the robotic arm's base.

$$\vec{X} = FK(\vec{q}) = \boldsymbol{T}_0^m = \prod_{i=1}^{m} \boldsymbol{T}_i^{i-1} \tag{11}$$

$$\vec{q} = FK^{-1}(\vec{X}) \tag{12}$$

$$\boldsymbol{T}_i^{i-1} = \begin{bmatrix} \boldsymbol{R} & \vec{d} \\ 0 & 1 \end{bmatrix} \tag{13}$$

The joint configurations of the 9-DOF and 10-DOF robotic systems are shown in figures 11 and 12 respectively. For both systems, rotations of each joint were found in order to model the kinematics of the manipulators. These rotations are

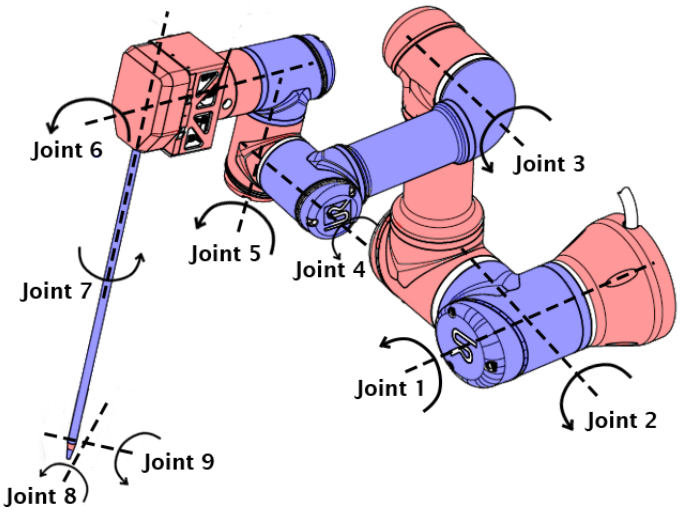


Figure 11: Joint rotations of 9-DOF robotic system comprised of 6-DOF UR3 serial manipulator and 3-DOF surgical tool

listed as *roll* (i.e a rotation around the x-axis), *pitch* (i.e. rotation around the y-axis) and *yaw* (rotation around the z-axis) in table 4.1.1. The forward kinematics of the manipulators were solved using the Euclidean transformation method as explained above. The inverse kinematics of the manipulators were solved using pseudoinverse of the Jacobian matrix. The calculation of the Jacobian matrix is further explained in the next subsection 4.1.2 and the inverse kinematics method is discussed in subsection 4.1.3.

4.1.2 Jacobian Matrix Calculations

For actuated and under-actuated manipulators, it is possible to find an analytical solution to the inverse kinematics problem. However, since in kinematically redundant manipulators, there exists an infinite number of joint configurations for a given end-effector pose, analytical solutions for the inverse kinematics problem do not exist. Therefore, the most common approach to solving inverse kinematics is to use iterative optimization methods for the approximation of the solution.

The most popular way of approximating the inverse kinematics solution is the Jacobian inverse technique. The Jacobian matrix is defined as a matrix that contains the first-order derivatives of a vector function. Equation 14 shows the

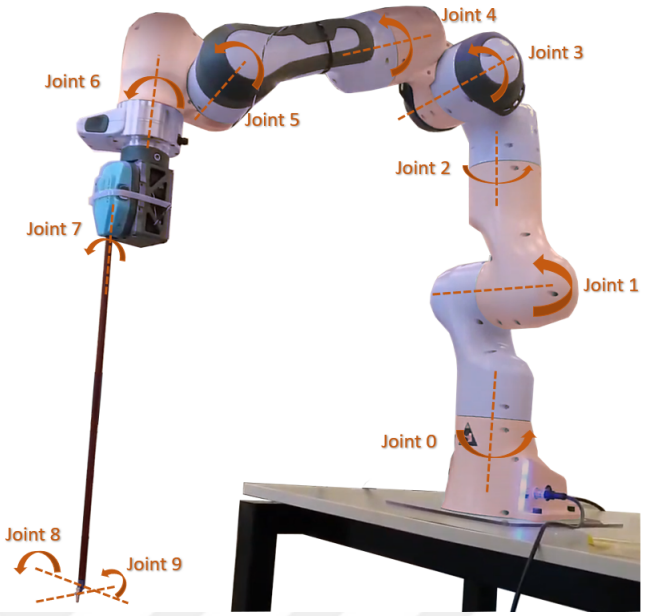


Figure 12: Joint rotations of 10-DOF robotic system comprised of 7-DOF Panda serial manipulator and 3-DOF surgical tool

Joint Number	Rotations	
	9-DOF w/ UR3	10-DOF w/ Panda
1	Roll	Yaw
2	Pitch	Pitch
3	Pitch	Yaw
4	Pitch	Pitch
5	Yaw	Yaw
6	Roll	Pitch
7	Yaw	Yaw
8	Pitch	Yaw
9	Roll	Pitch
10	-	Roll

Table 1: The rotations of each joint of the robotic systems

Jacobian matrix of $f(x)$ where f takes inputs from an n-dimensional space and produces an output vector in m-dimensional space. Thus, the resulting Jacobian matrix is defined to be mxn matrix.

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial f_1}{x_1} & \cdots & \frac{\partial f_1}{x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{x_1} & \cdots & \frac{\partial f_m}{x_n} \end{bmatrix}, \; where \; f : \mathbb{R}^n \to \mathbb{R}^m \tag{14}$$

For an inverse kinematics approximation of a manipulator, the Jacobian matrix is calculated by taking the partial derivatives of the pose function by each of the joint parameter. Therefore, a Jacobian matrix of an m-DOF manipulator is defined as 6xm matrix where each row of the matrix is the partial differentiation of each pose value shown in (10) by the joint parameters shown in (9). In a way, the Jacobian matrix of a manipulator can be thought as a way of mapping the joint space into the cartesian space as shown in (15).

$$\dot{\vec{X}} = \boldsymbol{J}\dot{\vec{q}} \tag{15}$$

Since tele-operation of a robotic arm requires fast calculations and the Jacobian matrices of the redundant manipulators are costly to calculate because of the amount of parameters, the Jacobian matrices are pre-calculated symbolically and saved as a function for the two manipulators that are used in this thesis. Therefore, the amount of calculations, that are made inside the tele-operation loop during the inverse kinematics, is minimized.

4.1.3 Inverse Kinematics

The Jacobian matrix of a manipulator approximately maps the joint space to the cartesian space. If the both sides of the equation (15) is multiplied by the inverse of the Jacobian matrix, then the cartesian space can also be mapped to the joint space, which is what the inverse kinematics approach is built on. For a 6-DOF robot, its Jacobian matrix is invertible as long as it has full rank, since

it is a 6×6 square matrix. However, for under-actuated or redundant systems, the Jacobian matrix is not invertable since it is not a square matrix. Therefore, pseudoinverse of the Jacobian matrix can be used instead. In equation (16), the Jacobian inverse technique for the inverse kinematics approximation as well as the right Moore-Penrose inverse of the Jacobian matrix is shown.

$$\dot{\vec{q}} = \boldsymbol{J}^{\dagger}\dot{\vec{X}} \ , where \ \boldsymbol{J}^{\dagger} = \boldsymbol{J}^{T}(\boldsymbol{J}\boldsymbol{J}^{T})^{-1} \tag{16}$$

Since the Jacobian inverse method is an iterative approximation method, the steps between the inputs must be sufficiently small. Even with small step iterations, *drifting* may occur in the system. The equation (16) alone is an approximation system, in which the error in the calculations accumulates over time. Therefore, a forward kinematics based system that corrects possible errors in the resulting joint parameters was implemented. The pose of the resulting joint angles from (16) was calculated using forward kinematics. Then the error between the target poses (i.e. X_{target}) and the current poses (i.e. $X_{current}$) was found and multiplied with an empirical constant k_{error} in (17). These errors in the cartesian space is converted to joint space using the Jacobian matrix in (18). Then the errors in the joint parameters are added to previous joint outputs to find the finalized joint parameters of the inverse kinematics approximation in (19).

$$\vec{X}_{error} = k_{error}(\vec{X}_{target} - \vec{X}_{current}) \tag{17}$$

$$\vec{q}_{error} = \boldsymbol{J}^{\dagger}\vec{X}_{error} \tag{18}$$

$$\vec{q}_{final} = \vec{q}_{output} + \vec{q}_{error} \tag{19}$$

Figure 13 shows the control structure of the inverse kinematics approximation of the system with an error correction structure that uses a P-control like system along with the forward kinematics that prevents drifting on the system. The previous pose and the previous angles of the manipulator are kept for the computations of $\dot{\vec{X}}$ and $\dot{\vec{q}}$ values in the (16). Then, the resulting joint parameters

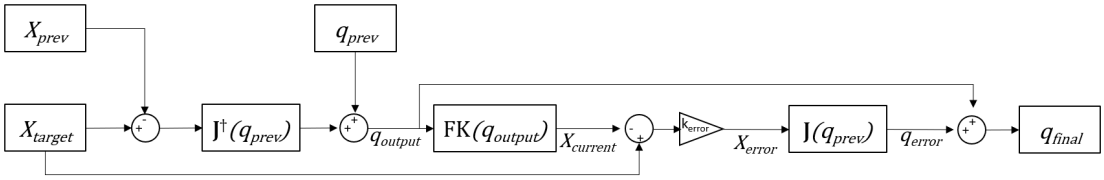


Figure 13: Control structure of conventional inverse kinematics with a forward kinematics based error correction system

are corrected by calculating the error between the target pose and the actual pose of the manipulator.

4.2 Adaptive Inverse Kinematics

During a robot-assisted surgery, fine and precise movements are essential. However, if the surgeon desires to cover a wider workspace, the system that is designed for fine movements may restrict the surgeon and prolong the duration of the operation. This can be avoided by changing the kinematics configuration of the manipulator according to the user's movements so that the robotic arm is capable of both fine movements and moving in wider workspaces. In this section, a work that aims to automatically switch in between different kinematics configurations according to the speed of the user's movements and the current manipulability of the robotic arm is explained.

In a robotic system, redundancy is defined as having extra freedom than needed in order to satisfy the target position and orientation of the end-effector. In a redundant system, the manipulator can reach the desired end-effector pose in infinitely many joint configurations since the inverse kinematics of the manipulator has infinitely many solutions. However, by locking certain joints at certain times, the manipulator can be restricted to be 6 degrees-of-freedom at all times so that the inverse kinematics can have a unique solution. By choosing which joints to be active at what times, the speed of the manipulator can be adjusted automatically and the efficacy of the teleoperation can be improved.

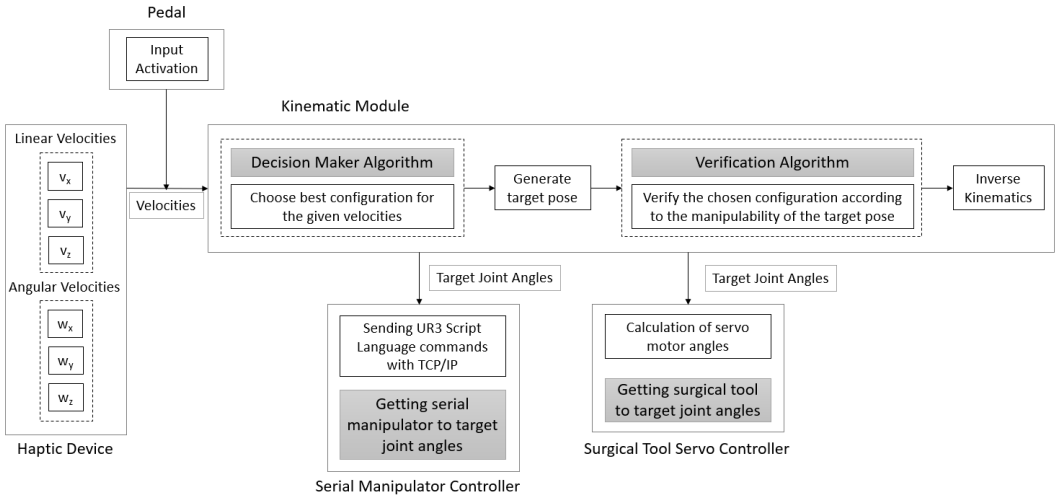


Figure 14: Data flow diagram of the complete system where the linear and angular velocities from the haptic device is sent to kinematics module if the pedal is pressed. The kinematics module then decides which kinematics configuration is suitable for the motion and calculates the joint angles.

4.2.1 System Setup

In order to implement the proposed methods, a 9 degrees-of-freedom (DOF) robotic system comprised of a 6-DOF serial manipulator (Universal Robots UR3, Denmark) and a 3-DOF servo-driven surgical tool (Intuitive Surgical Endowrist Needle Driver, USA) has been used as represented in Figure 11. The 3-DOF surgical tool is controlled with 4 Dynamixel XL-320 servo motors. The control scheme of the surgical tool and the manipulator is explained in more detail in Chapter 3.

The robotic system was manipulated with a 6-DOF haptic device (Force Dimension omega.6) and a pedal connected as a joystick module that helps with clutching. Communication between the instruments is attained with RS232, TCP/IP and Robot Operating System (ROS). The data flow of the complete system is shown in Figure 14 and the ROS map that shows the node and topic connections of the system is shown in Figure 15.

Since the workspace of the haptic device is smaller than the workspace of the manipulator, the user might have to readjust the pose of the haptic device without moving the robot. This operation is called clutching. In order to achieve this, a

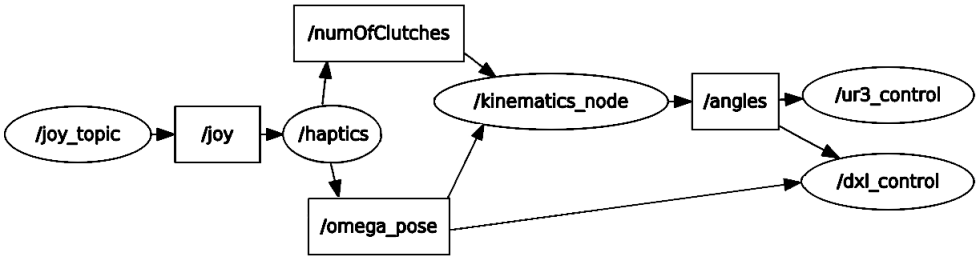


Figure 15: rosnode and rostopic map of the system where the oval shapes represent the nodes and the rectangular shapes represent the topics.

pedal is connected to the main computer as a joystick module. The pedal acts as an input enabler where the inputs from the haptic device are sent to the kinematics module if and only if the pedal is pushed by the user. Therefore, if the pedal is not used, user can freely move the haptic device without sending any signals to the manipulator.

The linear and angular velocities of the haptic device are sent to the kinematics module where the kinematics module decides which set of active joints would be suitable for the given velocities and the current manipulability of the robot. The joint angles are calculated according to the chosen kinematics configuration and sent to the serial manipulator controller of UR3 and the servo controller of the surgical tool.

4.2.2 Choosing Kinematics Configurations

The inverse kinematics of the system was calculated by using the cartesian linear and angular velocities of the haptic device. The Jacobian matrices for each chosen kinematics configuration was calculated and the end effector pose of the manipulator was iterated by scaling the haptic device velocities to manipulator's end effector. Because the transformation varies according to the pose of fixed joints at the moment they were fixed, the forward kinematics of the robotic arm was always calculated for all nine joints. However, the Jacobian matrices were calculated according to the different joint configurations.

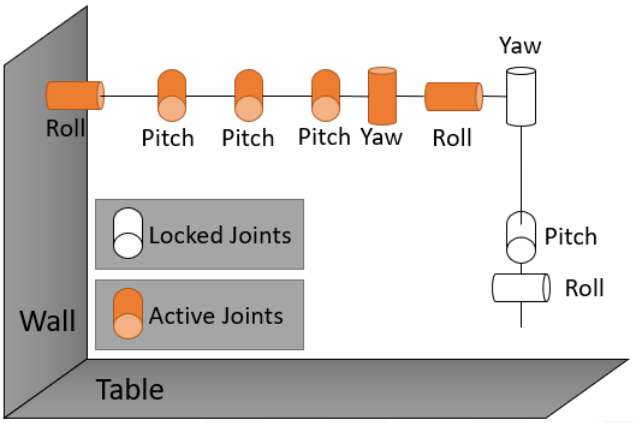
The system was assumed to ignore the remote center of motion (RCM) as it is

designed to operate on the surface, however the method is applicable to a surgical system that has RCM as long as the manipulator is redundant above RCM point.

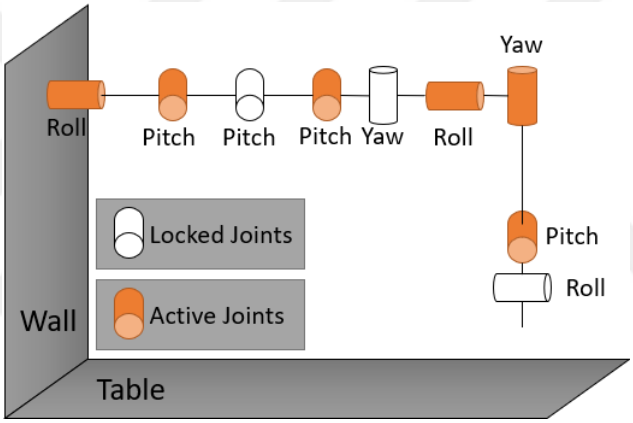
For adjusting end-effector speed of the 9-DOF system, 3 different kinematics configurations were chosen where the six out of the nine joints will be active at any given time so that the manipulator acts like a non-redundant one. It is observed that some pairs of joints would create similar rotational output. Therefore, while choosing the kinematics configurations, joints whose motion would create similar effects on the end effector were chosen.

In simple terms, manipulability analysis can be explained as the measure of manipulating ability of a robot at a given state. Before practicing the proposed methods on the physical system, empirical investigations were carried out in a simulated environment to find a suitable set of joint configurations that would be used as potential sets of active joints. Firstly, the inverse kinematics of various joint configurations were calculated and tested with random pre-set trajectories to see if it was flexible enough to follow different trajectories. If the joint configuration is good enough to follow given trajectories, the manipulabilities of these joint configurations were calculated by using (20). As a result of these empirical experiments, the three configurations that can scale the velocity of the end-effector the best were chosen. The chosen kinematic configurations are shown in figure 16. For the fastest configuration (figure 16(a)), all of UR3 joints were chosen to be active and all of surgical tool joints were chosen to be fixed since the largest end-effector motion was supplied via UR3 joints. For the medium (figure 16(b)) and slowest (figure 16(c)) configurations, some of the UR3 joints were also locked for scaling the end effector velocity. Further empirical experiments were carried out in the simulated environment to make sure switching in between these configurations does not cause any spikes on the pose and/or the angle values.

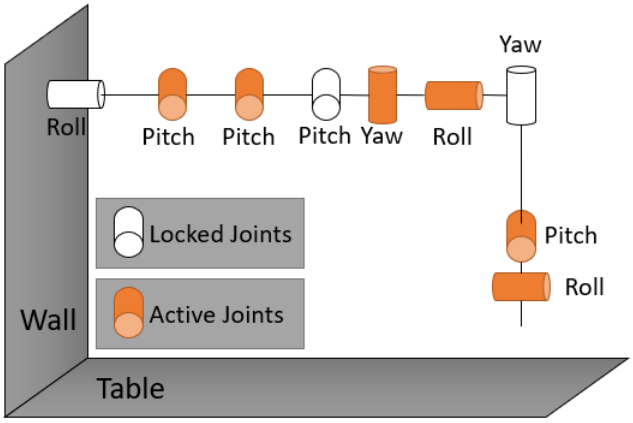
As mentioned earlier, manipulability of each joint configuration was analyzed before they were chosen to be used for motion scaling. As an example of this analysis, the manipulability ellipsoids of the chosen kinematics configurations were



(a) Fastest Configuration



(b) Medium Configuration



(c) Slowest Configuration

Figure 16: Locked and active joints of the chosen kinematics configurations

drawn when the robotic arm was at the home position (shown in figure 17). In these figures, long axes of the ellipsoids represent the direction where the end effector can move most quickly. Manipulability constant can take a value between zero and one where having a zero manipulability indicates that the robotic arm is in singular position and having a manipulability constant of one indicates that the workspace of the robotic arm is in maximum capacity. Manipulability constants of the chosen configurations at the home pose are 0.1540, 0.1087, and 0.0156 respectively from the fastest configuration to the slowest.

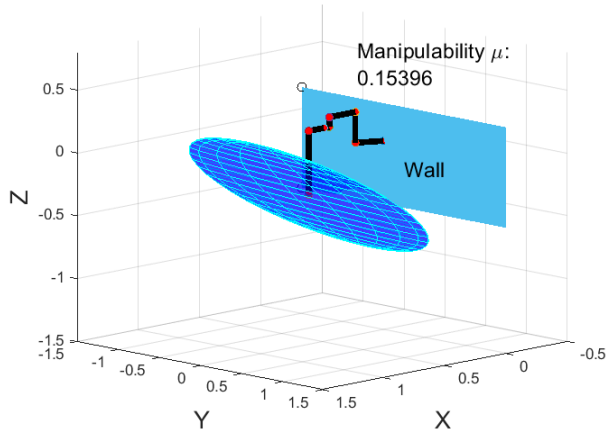
$$\mu = \sqrt{det(\boldsymbol{J}\boldsymbol{J}^{\boldsymbol{T}})} \tag{20}$$

4.2.3 Switching the Configurations

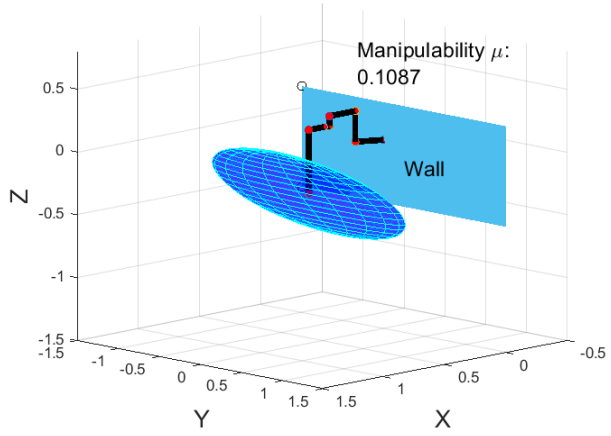
For automating the switches between the three kinematics configurations, a decision making algorithm has been implemented that chooses the best kinematics configuration according to the user motion (Algorithm 1). However, this algorithm does not consider singularity cases. Thus, when the kinematics configuration is switched, the robotic arm might be in near singular pose where it would not be if another set of joints would have been active. Therefore, in order to avoid singularity at all costs, another algorithm has been implemented to switch the kinematics configuration again in case of singularity (Algorithm 2). The singularity avoidance algorithm tries to find a configuration that will not put the manipulator in a near singular pose. If the manipulator is in singularity for all possible active joint sets for the user's current motion, then it prevents the user to make that motion.

Both algorithm 1 and 2 are iterated every time data is received from the haptic device. Algorithm 1, first finds out whether the linear or angular motion is more dominant on the user's motion. From there, it finds the most dominant axis that the user intends to move. After finding the most dominant movement, it decides which kinematic configuration would suit the best for the given velocity.

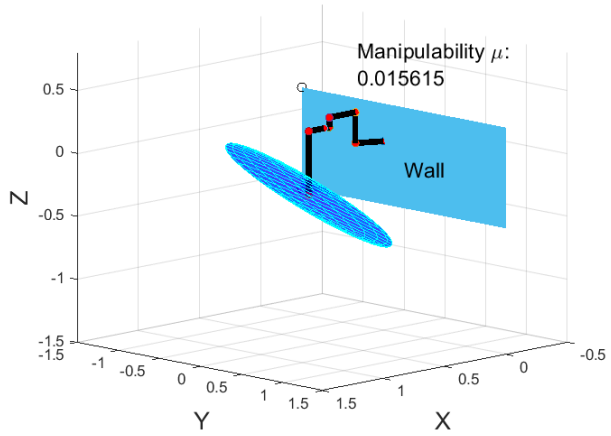
In some cases, manipulability of the robotic arm decreases significantly in



(a) Fastest configuration $\mu = 0.1540$



(b) Medium configuration $\mu = 0.1087$



(c) Slowest configuration $\mu = 0.0156$

Figure 17: Manipulability ellipsoids and constants of the chosen three configurations at robot's home pose. Long axes of the the ellipsoids are the direction where the end-effector of the robot can move most quickly.

slower configuration even though it is higher for faster joint sets. Therefore, algorithm 2 needs to verify the joint configuration chosen by algorithm 1 to make sure the target position is acquirable by the selected configuration. The algorithm checks the manipulability constant of the selected joint configuration and if the manipulability constant is smaller than the pre-determined value of 0.015, the case is switched to a faster kinematic configuration. This operation is repeated if necessary, until the manipulability constant is an acceptable value or if it is not acceptable for even the fastest configuration, the algorithm does not allow the user to make the said movement in that direction, therefore forcing the user to avoid singular poses.

Algorithm 1 Kinematics Decision Maker

Require: $\nu_x, \nu_y, \nu_z, \omega_\alpha, \omega_\beta, \omega_\gamma$: Linear and angular velocities from the haptic device

$\bar{\nu} \leftarrow average\,of\,\nu_x,\ \nu_y,\ and\,\nu_z$
$\bar{\omega} \leftarrow average\,of\,\omega_\alpha,\ \omega_\beta,\ and\,\omega_\gamma$
$pos_{decision} \leftarrow true$
if $(\bar{\nu} < K_{lower})\,or\,(\bar{\omega} > K_{upper})$ **then**
    $pos_{decision} \leftarrow false$
end if
if $pos_{decision}$ **then**
    $dominant \leftarrow Find\,dominant\,axis\,of\,\nu_x,\ \nu_y,\ \nu_z$
    if $dominant > \nu_{config\,1}$ **then**
        Set case to fastest configuration
    else if $dominant > \nu_{config\,2}$ **then**
        Set case to medium configuration
    else
        Set case to slowest configuration
    end if
else
    $dominant \leftarrow Find\,dominant\,axis\,of\,\omega_\alpha, \omega_\beta, \omega_\gamma$
    if $dominant > \omega_{config\,1}$ **then**
        Set case to fastest configuration
    else if $dominant > \omega_{config\,2}$ **then**
        Set case to medium configuration
    else
        Set case to slowest configuration
    end if
end if

Algorithm 2 Configuration Verification Algorithm

Require: *goalPose* : Destination pose of the robot, *case* : Selected kinematics configuration by Algorithm 1

```
μ ← manipulability constant for goalPose for config case
while μ < 0.015 do
    if case == slowest then
        Set case to medium
        Update μ for medium configuration
    else if case == medium then
        Set case to fastest
        Update μ for fastest configuration
    else
        Robot is in singular position for all configurations
        Break
    end if
end while
```

4.2.3.1 Joint Velocity Analysis at Configuration Switches

In order to observe the velocity spikes of the joints that occur when a configuration switch takes place, an example trajectorty with a linear velocity that decrements over time from 0.15 m/s to -0.15 m/s has been created. Figure 18 shows the velocity changes at each joint during this example trajectory. The dashed red lines on each subfigure indicates a configuration change at that time step. The kinematic configuration starts with the fastest one, then switches to medium, then to the slowest configuration. After that, as the absolute velocity of the end-effector increases again, it switches back to medium configuration and finally it finishes off with the fastest configuration.

From analyzing this trajectory, biggest trajectory spike occurs at joints 1 and 3 at 0.16 rad/s. As it was previously explained in subsection 3.2.1, a special command for online control of the UR3 robot (i.e. *servoj*) has been used. This command predicts the future position of the manipulator using a parameter, look ahead time, and brings the joints to the target angles without changing the velocity too much. Therefore, the spikes in the joint velocities are small enough to be compensated by the robot controller, hence the effect of these discontinuities will not be observed in the real life system.

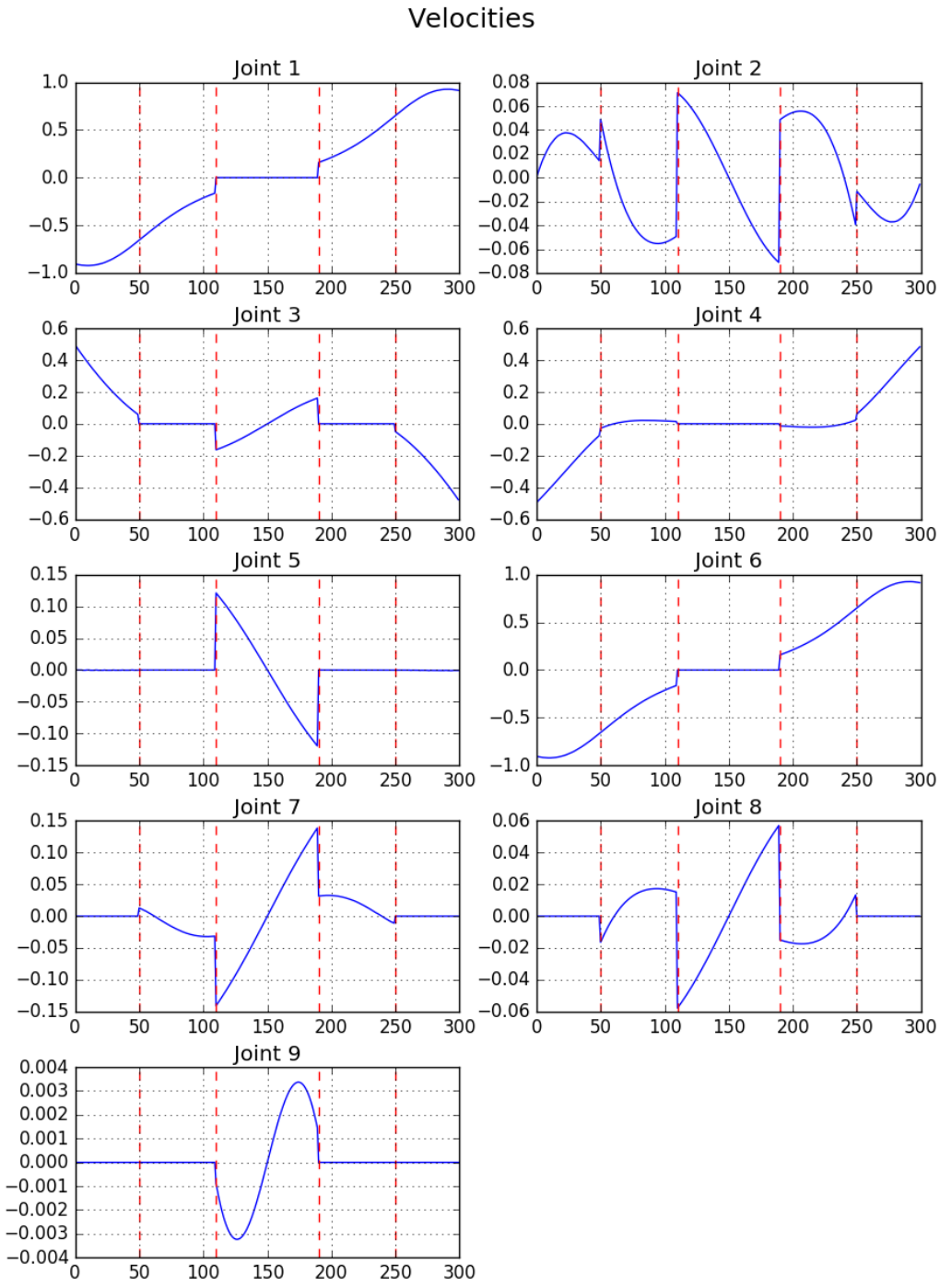


Figure 18: Velocity spikes of each joint from kinematic calculations when configuration switches. The velocity of each joint is plotted when the velocity of the end-effector is slowly decremented from 0.15 m/s to -0.15 m/s. Dashed red lines indicate a configuration switch at that time step.

4.2.4 Experimental Results

In order to verify the proposed method, a peg and ring experiment was designed for inexperienced users to carry out. Ten subjects who have never used the robot were asked to take different colored plastic rings with different girths from the starting peg to their corresponding same colored pegs. The experiment setup with 9-DOF manipulator, 6-DOF haptic device and the peg board has been shown on figure 19. All ten subjects has been asked to complete the same task twice with both redundant (i.e. standard 9-DOF inverse kinematics) and the proposed adaptive system. Half of the subjects used the adaptive system and the other half used the redundant system first in order to eliminate the affect of learning on the experimental results. Before the experiments are started, two out of five rings were used as practice so that the subjects can get used to the control of the robot. For each experiment, the duration of the task and the number of clutches are kept as the resulting data. For adaptive kinematics experiments, the switches between the configurations were also recorded. Figure 20 shows the configuration change throughout the time of one experiment of a randomly selected user. On this graph, yellow lines labelled as Config 2 represent the slowest configuration, green lines labelled as Config 1 represent the medium configuration, and the purple lines labelled as Config 0 represent the fastest configuration.

The mean and standard deviation ($\mu \pm \sigma$) of the number of clutches during the redundant kinematics are 69.5±16.5. During the adaptive kinematics means and the deviation of the number of clutches were 52.5±13.8. The mean and standard deviation of the redundant kinematics experiment duration were 352.6±104.4 seconds, and the mean and standard deviation of the adaptive kinematics duration were 268.3±71.5 seconds. Null-hypothesis of having no difference between the adaptive and redundant kinematics has been rejected by applying t-tests to the time and clutch data pairs of the experiment ($p_{duration} = 0.0021$, $p_{clutch} = 0.0005$). Results of the collected data are represented in Fig. 21(a) and 21(b). Also after the experiments were completed, the subjects were asked which kinematic scheme

Figure 19: Experimental setup: 9-DOF surgical robotic arm, 6-DOF haptic device, and the task given to the subjects

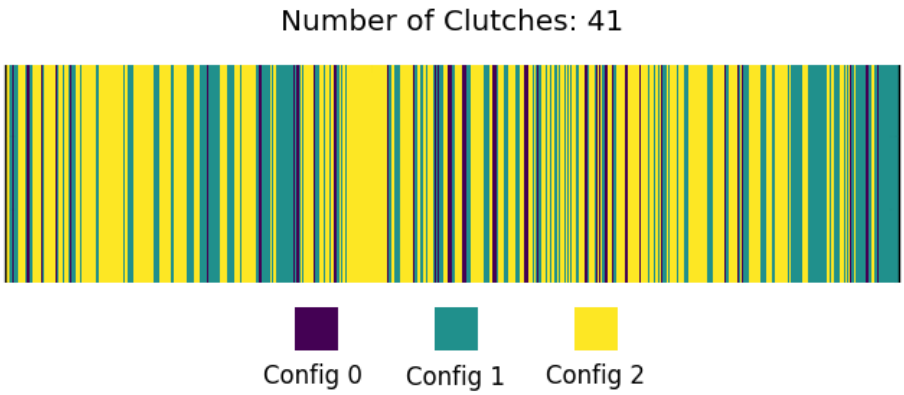
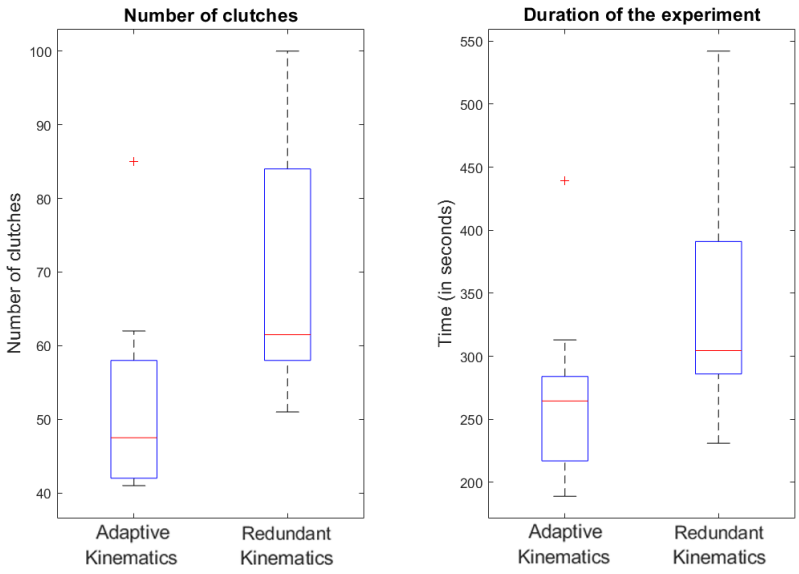


Figure 20: Configuration switches throughout one experiment of a randomly chosen subject with number of clutching done reported on top of the graphç



(a) Number of Clutches (b) Duration of the Experiment

Figure 21: Experiment results: comparison of adaptive and redundant kinematics across ten subjects. There is significant difference between the adaptive and redundant kinematics.

was more comfortable to control, 9 out of 10 subjects stated that they preferred the adaptive kinematics over the redundant kinematics. Only one subject thought the redundant kinematics was easier to control even though their performance was better in adaptive kinematics case.

4.3 Haptic Force Cues

When a tele-operator is performing a task on a manipulator, they might not realize the physically non-optimal poses of the robotic arm such as singularity or proximity to joint limits. The objective of this section of the thesis is to alert the user in case of a non-optimal pose on the manipulator using the directional force cues through the haptic device as the interaction tool between the user and the robotic arm. When the manipulator approaches to a proximity of a singular pose or to a limit of any joint, a scaled force cue is given to the user, by the haptic device, towards the direction of where the manipulator can escape from the non-optimal pose. Doing so, it is aimed to reduce the cognitive load of the tele-operator and help them focus on their tasks.

A problem with a tele-operation system with any kind of force feedback is that it might be physically tiring for the tele-operator as the duration of the task increases. Therefore, the scaling of these forces must be tuned such that the tele-operator is not burdened with too much force. For the case of this study, there is no need to apply a force feedback on the haptic device constantly. The user only needs to be warned in case of a non-optimal pose and when close to a non-optimal pose. If the manipulator is far away from such cases, the force cues can be negligible so that the manipulation is not tiring for the tele-operator.

For the tests of this study, a 10-DOF robotic system comprised of a 7-DOF Panda serial manipulator and a 3-DOF surgical tool that was shown in figure 12. For the manipulation of the robotic system, Touch haptic device (figure 8) and two pedals, one for clutching and one for switching the rotation input of the haptic device from whole system orientation to surgical tool orientation, have been used. Inverse kinematics of the robotic system was solved using the conventional kinematics methods as explained in section 4.1. For the 7-DOF Panda manipulator's control scheme, joint impedance control interface has been used (see section 3.2). The 3-DOF surgical tool's servo motors were actuated using the methods that were explained in section 3.4. Finally, the velocity inputs were received from the haptic device as explained in section 3.3.

4.3.1 Cost Functions

In order to scale the robotic arm's proximity to a non-optimal pose, two cost functions were constructed and summed to create an overall cost function of the system. Both cost functions use an exponential structure with a quadratic power so that the cost gets higher exponentially as the manipulator gets closer to the non-optimality, but it does not have so much effect if the manipulator is far from the non-optimality. As mentioned before, giving a constant force feedback to a tele-operation system can be physically tiring for the tele-operator. Therefore, these cost functions were constructed in such a way that the cost is negligible if the manipulator is far away from the non-optimality (i.e. joint limits and singularity).

Joint Number	q_{min} **(rad)**	q_{max} **(rad)**
Joint 1	-2.8973	2.8973
Joint 2	-1.7628	1.7628
Joint 3	-2.8973	2.8973
Joint 4	-3.0718	-0.0698
Joint 5	-2.8973	2.8973
Joint 6	-0.0175	3.7525
Joint 7	-2.8973	2.8923
Joint 8	-2.3561	2.3561
Joint 9	-1.5708	1.5708
Joint 10	-1.5708	1.5708

Table 2: The joint limits of the 10-DOF system comprised of 7-DOF Panda robot and 3-DOF surgical tool

In order to make sense of the manipulator's proximity to the joint limits, a cost function that utilizes the limits of each joint that are shown on table 2 has been constructed as in (21). The resulting cost, $h_j(\vec{q})$, must be zero when all the joints are at the middle point of the upper and lower limits of each corresponding joint, and should increase exponentially as the joint angles approach to lower or upper limits. Equation 21 calculates the cost of each joint individually and later sums them up to find the final $h_j(\vec{q})$.

Similar to the joint limits cost function, another cost function, shown in (22), has been constructed by using the manipulability index μ, for the manipulator's proximity to a singular pose. Manipulability index, μ, is defined as a measure of a manipulator's ability to move the pose of its end-effector. The singularity cost, $h_s(\vec{q})$, should be zero when the manipulability of the end-effector is at maximum capacity and should increase exponentially as the manipulability index decreases (i.e. the manipulator closes to a singular pose).

$$h_j(\vec{q}) = \alpha_j \sum_{i=1}^{n} [\exp\left(\beta_j (q_i - q_{i,min})^2\right) + \exp\left(\beta_j (q_{i,max} - q_i)^2\right)] \tag{21}$$

$$h_s(\vec{q}) = \alpha_s \exp\left(\beta_j \mu^2\right), \ where \ \mu = \sqrt{\det(\boldsymbol{J}^T \boldsymbol{J})} \tag{22}$$

$$H(\vec{q}) = h_s(\vec{q}) + h_j(\vec{q}) \tag{23}$$

The overall cost function for the non-optimality of the manipulator is found by using (23) which is the sum of joint limits cost function (21) and singularity cost function (22). The α_j, α_s, β_j, and β_s constants of the (21) and (22) functions should be tuned in such a way that the finalized cost function, (23), is affected in the same scale with both joint limit costs and singularity costs. Therefore, neither reaching the joint limits nor nearing to a singular pose has dominance on the resulting cost of the robotic arm's current configuration.

4.3.2 Force Calculations

The force cues for the haptic device are calculated using the cost function given by (23). Equation (24) finds the set of angles that would minimize the overall cost with the term $\nabla_{\vec{q}}H$ and these angles were converted to the cartesian space with Jacobian matrix. Then the forces, $\vec{F}$, for the haptic device were scaled using the k_f constant.

$$\vec{F} = k_f(\boldsymbol{J}^{\dagger})^T \nabla_{\vec{q}}H \tag{24}$$

The set of angles that would minimize the cost function were calculated by multiplying the cost difference with the partial derivative of the cost function with respect to the angles as shown in (25). Similar to the calculation of the Jacobian matrix for the inverse kinematics approximation, the partial derivative of the joint limits cost function was calculated symbolically and saved as a function to reduce the computation time. The partial derivative of the singularity cost function is calculated using (26). The calculations for the partial derivative of the manipulability index was further explained on the subsection 4.4.2.

$$\nabla_{\vec{q}}H = \frac{\partial H(\vec{q})}{\partial q}\Delta H(\vec{q}) = \left(\frac{\partial h_j(\vec{q})}{\partial q} + \frac{\partial h_s(\vec{q})}{\partial q}\right)\Delta H(\vec{q}) \tag{25}$$

$$\frac{\partial h_s(\vec{q})}{\partial q} = 2\alpha_s\beta_s\frac{\partial \mu}{\partial q}\exp\left(\beta_j\mu^2\right) \tag{26}$$

The resulting forces, $\vec{F}$, from the (24), were sent to the haptic device which gives directional force cues for the user in three dimensions, x, y, and z, which would free the manipulator from a non-optimal configuration.

4.4 Passive Manipulability Maximization

As defined in section 4.1 previously, a redundant manipulator can reach a certain point in the cartesian space with an infinite number of joint configurations. Since the Jacobian inverse method is an iterative method, the new joint parameters are calculated using the previous joint parameters. Therefore, inverse kinematics finds a solution for the target pose of the manipulator based on its previous status. However, in a redundant robotic arm, the joint configuration found by the Jacobian iteration might not be the most optimal solution. The objective of this section of the thesis is to maximize the manipulability of the robotic arm passively, by exploiting the redundancy, without interfering with the position and orientation of the end-effector.

For the tests of this study the same setup has been used with the section 4.3. The 10-DOF robotic system with 7-DOF Panda Manipulator and a 3-DOF surgical tool (figure 12), was manipulated using a 6-DOF Touch serial haptic device (figure 8) and two pedals. However, for the tele-operation with haptic force cues, the inverse kinematics and haptic force cue calculations were only being made if the clutching pedal was pushed. In the passive manipulability maximization case, the inverse kinematics calculations were resumed, since the kinematics can always find a better joint configuration even if the robotic arm's end-effector was not moving.

The manipulability maximization design replaces the conventional kinematics as explained in the subsection 4.1.3. Therefore, the data flow of the system and error compensation feedback loop (figure 13) are structurally the same as the

conventional kinematics. The only change from conventional kinematics is the way that the inverse kinematics is being calculated.

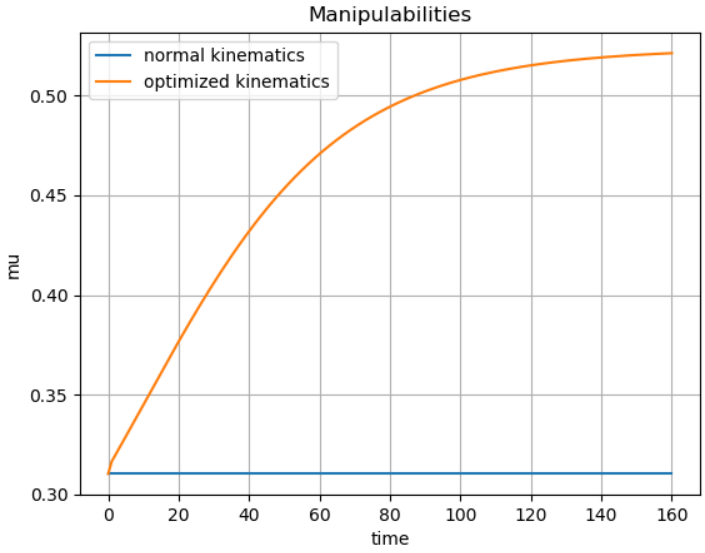
4.4.1 Nullspace of the Jacobian Matrix

The introduced method of passively increasing the manipulability of the robotic arm can be achieved by utilizing the internal motions of the joints. Therefore, the Jacobian based inverse kinematics method that was shown in (16) has been modified to (27) to include the nullspace motion of the manipulator to the inverse kinematics equation. The term $(\boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{J})$ finds the nullspace of the Jacobian matrix. The term $\vec{b}$ is the vector of a motion that will be conducted in the nullspace of the Jacobian matrix, thus in the nullspace of the robotic arm. Since the objective is to maximize the manipulability of the robotic arm, the vector $\vec{b}$ in (27) is replaced with the partial differentiation of the manipulability index μ with respect to joint parameters in (28).

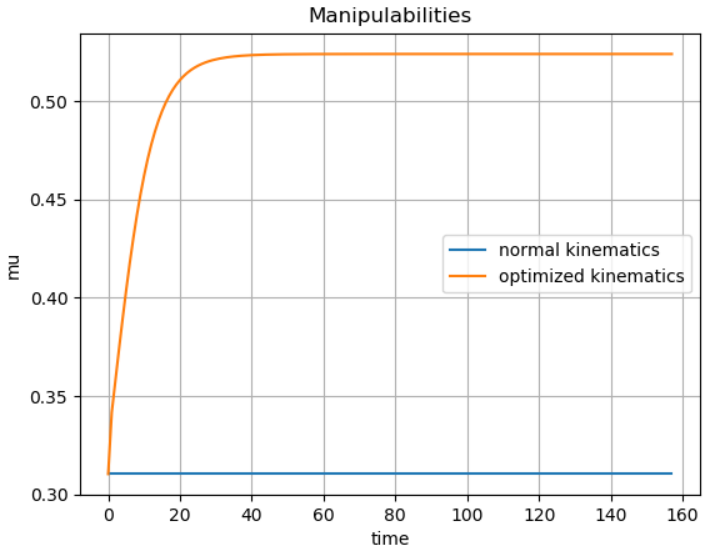
$$\dot{\vec{q}} = \boldsymbol{J}^{\dagger}\dot{\vec{X}} + (\boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{J})\vec{b},\ where\ \vec{b} \in \mathbb{R}^m \tag{27}$$

$$\dot{\vec{q}} = \boldsymbol{J}^{\dagger}\dot{\vec{X}} + (\boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{J})k\frac{\partial \mu}{\partial q} \tag{28}$$

The first part of (28) calculates the joint angles like the conventional kinematics. In the second part, partial differentiation of the manipulability index is used for increasing the manipulability of the end-effector in the nullspace of the robot. k is the scaling constant of how much this nullspace operation will be conducted, which means bigger the k is, bigger the internal motions of the joints are per iteration. With no new target poses incoming to a system, a system with a bigger k constant stabilizes sooner than a system with a smaller one. Figure 22 compares how fast would the system stabilize when no new input pose is coming to the kinematics module with different k constants. Figure 22(a) shows the case where $k = 0.1$, where figure 22(b) shows the case where $k = 0.5$.



(a) $k = 0.1$



(b) $k = 0.5$

Figure 22: Stabilization of the manipulability index when different k constants are used.

4.4.2 Manipulability Index Differentiation

Especially in a highly redundant system with a high dimensional Jacobian matrix, calculation of the manipulability index, shown in (29), can be very computationally costly. Therefore, taking the numerical differention of this index can slow down the system and the results may be inaccurate. Hence, similar to the Jacobian matrix calculations as explained in subsection 4.1.2, the partial differentiation of the manipulability index is precalculated symbolically and saved as a function.

$$\mu = \sqrt{\det(\boldsymbol{J}\boldsymbol{J}^T)} \tag{29}$$

The Jacobi's formula (30) states that, a partial differentiation of a matrix determinant is trace of the adjugate of the matrix times the partial differentiation of the matrix itself. This formula can be derived to the second part of (30) which removes the adjugate term and makes the calculation easier.

$$\frac{\partial \det \boldsymbol{A}(t)}{\partial t} = tr\left(adj(\boldsymbol{A}(t))\frac{\partial \boldsymbol{A}(t)}{\partial t}\right) = \det \boldsymbol{A}(t)\ tr\left(\boldsymbol{A}(t)^{-1}\frac{\partial \boldsymbol{A}(t)}{\partial t}\right) \tag{30}$$

Replacing the $\boldsymbol{A}$ and t values of (30) with the manipulability index μ and joint parameters q, equation 31 is reached. All of the terms in (31) can be calculated numerically in real time except for the partial differentiation of the Jacobian matrix. Therefore, the partial differentiation of the Jacobian matrix with respect to the joint parameters is calculated symbolically and saved as a function. Rest of the operation is conducted numerically in real-time.

$$\frac{\partial \mu}{\partial q} = \mu\ tr\left((\boldsymbol{J}\boldsymbol{J}^T)^{-1}\frac{\partial(\boldsymbol{J}\boldsymbol{J}^T)}{\partial q}\right) \tag{31}$$

4.5 Experiments with Haptic Force Cues and Manipulability Maximization

In order to validate and compare the results of the haptic force cue system, explained in section 4.3, and manipulability maximization system, explained in section 4.4, an experiment setup has been constructed. Data was collected from the

same experiment with six different setups:

1. Conventional kinematics
2. Haptic force cues using conventional kinematics where $k_f = 0.5$
3. Haptic force cues using conventional kinematics where $k_f = 0.75$
4. Haptic force cues using conventional kinematics where $k_f = 1.0$
5. Manipulability maximization
6. Haptic force cues using manipulability maximization

where k_f is the constant from the equation(24) which scales the amount of forces on the haptic device. These data are later used for answering four different questions that the studies have aspired to answer:

1. How do different levels of forces affect the quality of experiment?
2. Does giving back haptic force cues help user to avoid non-optimality of the robot?
3. How effective is the manipulability maximization compared to conventional kinematics?
4. Would it be more efficient in terms of the amount of forces if manipulability maximization is used with haptic force cues instead of conventional kinematics?

4.5.1 Experiment Setup

For the data collection, a simple peg-and-ring experiment has been designed as shown in figure 23. The user picks up the ring from the red peg and moves it to the purple peg using the Touch haptic device (see figure 8) to manipulate the 10-DOF robotic system comprised of 7-DOF Panda serial robotic arm and 3-DOF surgical tool (see figure 12). The difficulty of the task is kept simple to minimize the learning affect on the results. Ten relevant data samples are collected from each of the 6 setups that are listed above. For each of the setups, half of the data

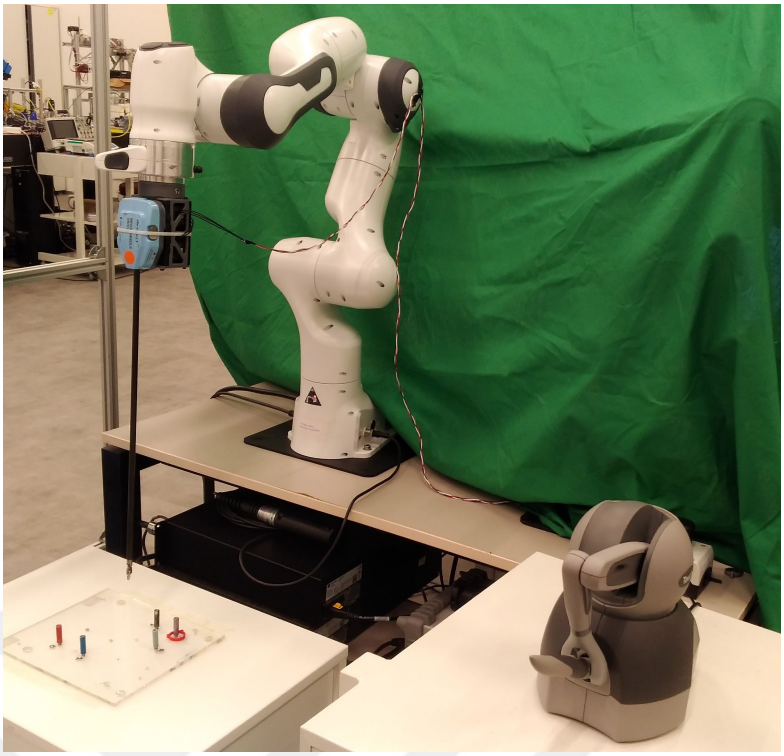


Figure 23: Experiment setup for the haptic force cue and manipulability maximization experiments: A peg and ring setup as a task on bottom left, the manipulator in the middle and the haptic device on bottom right

is collected first for each of them (i.e. five datasets for each setup), then the rest of them are collected in the reverse order. For example, if haptic force cues with $k_f = 0.5$ was the first experiment for the initial half, then it is moved to be the last experiment for the second half of the dataset. Thus, affect of learning on the results are further minimized.

The generalized dataflow of all experiments for this part is shown in figure 24. The user uses the haptic device and pedals to provide velocity inputs for a kinematics module. If the manipulability maximization is being used for the setup, the joint angles are calculated using manipulability maximization, otherwise conventional kinematics methods are used. The calculated joint angles are sent to the robotic arm's and the surgical tool's control schemes and robotic system is brought to the target angles. If the experiment uses the haptic force cue system, then the non-optimalities of the manipulator are checked and a force feedback is generated for the user through the haptic device by the haptic force cue generator.

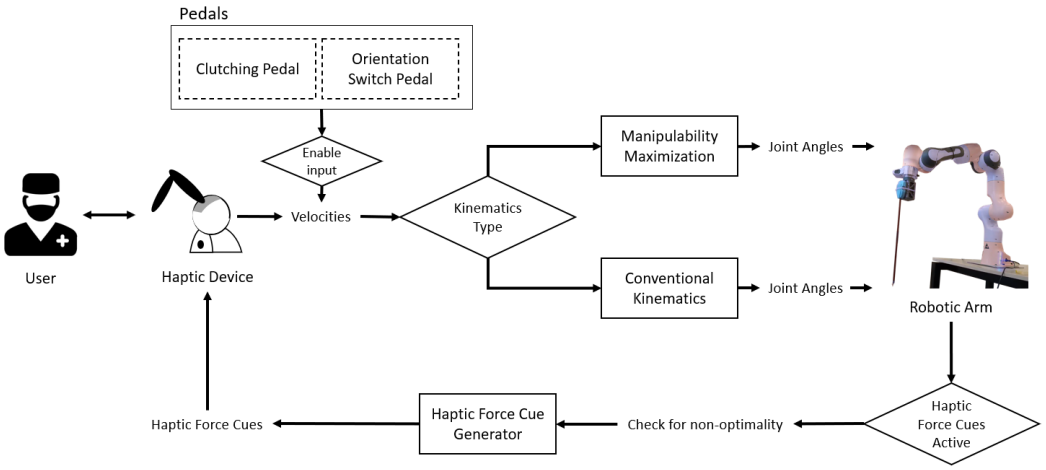


Figure 24: Dataflow of the haptic force cue generation and manipulability maximization experiments: The user sends goal points for the manipulator in cartesian space. The joint angles are calculated and sent to the robotic system by either the conventional kinematics or the manipulability maximization. Then, if the force cues are active, the force cue generator calculates and gives back haptic force cues for the user.

The joint angles of the robotic system are always brought to the same angles at the start of each experiment so that the results do not get affected by the starting configuration of the manipulator.

4.5.2 Comparisons of the Systems

As mentioned, four research questions are aimed to be answered in the design of the experiments that are explained in this section. For this purpose six different experiment setups are prepared. For the affects of the different amounts of forces, 3 different k_f constants (see equation 24) are used to scale the amount of output forces generated by the haptic force cue generation. These data derived from these values are also compared with conventional kinematics with no force feedback to answer the second question of if giving force feedback helps the user to avoid the non-optimal poses for the manipulator. The results for these two affects are presented in 4.5.2.1. Another investigation point of these experiments were to analyze the efficacy of the manipulability maximization. For this part, conventional kinematics and manipulability maximization is calculated for the same experiment, and the results are compared in 4.5.2.2. Since the manipulability maximization

Experiment Type	Completion Time (seconds)	% of time spent when cost >2	% of time spent when cost >2.5	Average Overall Costs
k = 0	37.67	37.45%	18.47%	2.0446
k = 0.5	35.00	28.89%	18.62%	1.9926
k = 0.75	31.58	25.99%	16.59%	1.9528
k = 1.0	27.08	23.82%	14.60%	1.9352

Table 3: Results of experiments conducted with different levels of forces

replaces conventional kinematics methods for the joint angles calculations, it can be used alongside of haptic force cue generation. 4.5.2.3, combines the manipulability maximization and haptic force cues to explore if the amount of forces can be made more efficient compared to using conventional kinematics.

4.5.2.1 Impacts of Different Levels of Forces

For determination of the affects from different levels of forces, four experiment setups are prepared. Firstly, no force feedback is given to the system (i.e. ordinary tele-operation with conventional kinematics). Secondly, k_f from the force generation equation, (24), is set to 0.5 which generates a minor but perceivable force feedback. Thirdly, k_f is set to 0.75 which generates a decent amount of force for the designed task, and finally it is set to 1.0 which might generate an overwhelming amount of force at some points during the task.

For all four of the experiments, joint costs, singularity costs, overall costs, manipulabilities, and the generated forces are collected as the data as shown in figure 25. From these data, some assumptions are made:

- The tele-operator would finish the task faster if the forces are higher.
- The tele-operator would avoid the non-optimal poses of the manipulator and therefore, spends less time in the configurations where the overall costs are higher.
- The average overall costs would be less in higher forces since the tele-operator spends less time in higher overall costs.

The average results of these experiments are shown in Table 3. Each row of the

table represents different experiment setups with different k_f constants, first row being no forces to last row being the maximum amount of forces. Each column shows the average results of ten trials for each setup. First column shows the average completion times, the second one shows the percent of time that is spent in configurations where the overall cost is higher than 2, the third one is for the same percentage for the overall cost of 2.5 and finally last column is for the average overall costs of the experiments. As the results on table 3 suggests, all of the assumptions that are listed above have been proven to be true.

4.5.2.2 Manipulability Maximization Effectiveness

In order to analyze and evaluate the efficacy of the manipulability maximization, both conventional kinematics and manipulability maximization are used for data collection during the experiment although only manipulability maximization is used as a source of input to the robotic system. The goal of this study was to optimize the manipulability of the end-effector by using the internal joint motions that transpires in the nature of redundancy, without moving the cartesian pose of the end-effector. Thus, the manipulability indices (see equation 29) are collected to see the quantity of the optimization on the manipulability. An example of the manipulability indice comparison between the conventional kinematics and the manipulability maximization is shown in figure 26. To observe the internal joint motions, the change in joint angles are also saved as shown in figure 27. Finally, for making sure that the end-effector pose actually stays the same during the internal motions of the joints, end effector poses are also saved as shown in figure 28. For all three figures, orange lines labelled as *optimized*, represents the results of manipulability maximization where the blue lines labelled as *normal* represents conventional kinematics (in the pose figure, the orange lines and blue lines are completely intersected).

As the figures 26, 27 and 28 reveal, the manipulability indices can greatly be improved using the techniques explained in 4.4. For better demonstrating the levels of optimizations of the method, average manipulability indices of each

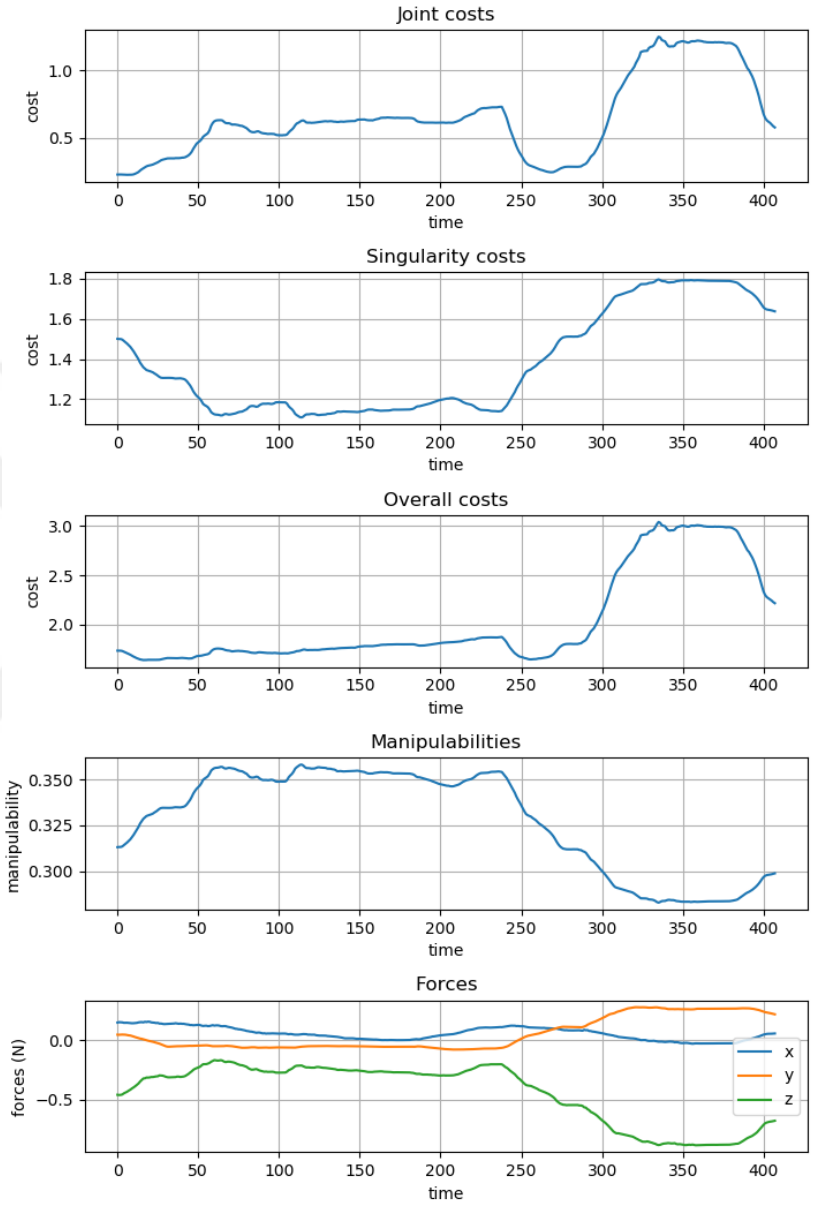


Figure 25: Example data from the haptic force cue experiments from one iteration of the peg and ring task. The plots show the results of the joint costs, the singularity costs, the overall costs, the manipulabilities and finally the calculated forces in three dimensions during one experiment by time step.

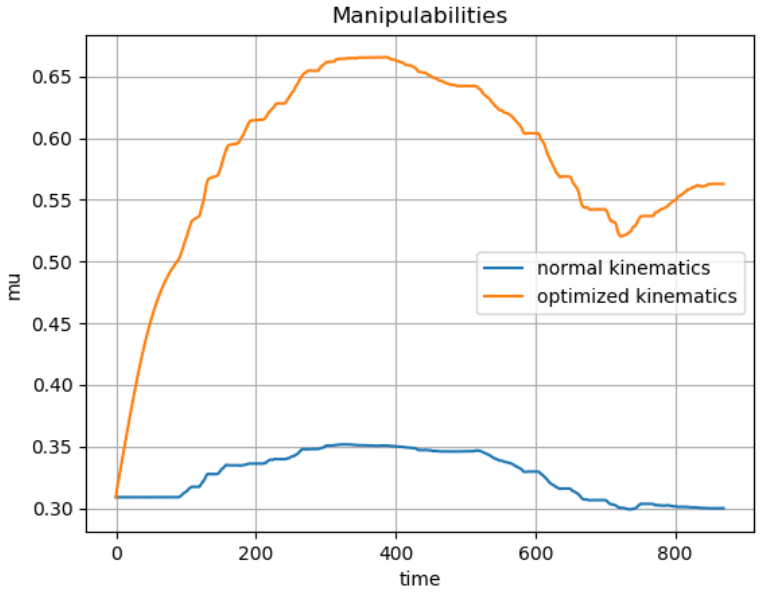


Figure 26: The manipulability difference between conventional kinematics and manipulability maximization during one experiment

experiment are calculated for both conventional kinematics and manipulability maximization and are shown on table 4. Overall average manipulability of the manipulability index of the manipulability maximization method is found 0.5726, where the average manipulability of the conventional kinematics is found 0.3237, which means the manipulability index of the utilized robotic system was found to be 56.53% more effective when compared with conventional methods for this specific experiment.

4.5.2.3 Haptic Force Cues with Manipulability Maximization

As the mathemetical structure, the manipulability maximization method replaces the conventional inverse kinematics methods. Therefore, it can be used as a kinematics module as well. Since the haptic force cue generation also depends on the manipulability index of the end effector, this part explores if the amount of output forces generated from the haptic force cue module can be optimized if manipulability maximization module were to be used instead of conventional kinematics.

For the experiments of this part, the conventional kinematics is switched to the manipulability maximization and same data as the 4.5.2.1 are collected. The newly

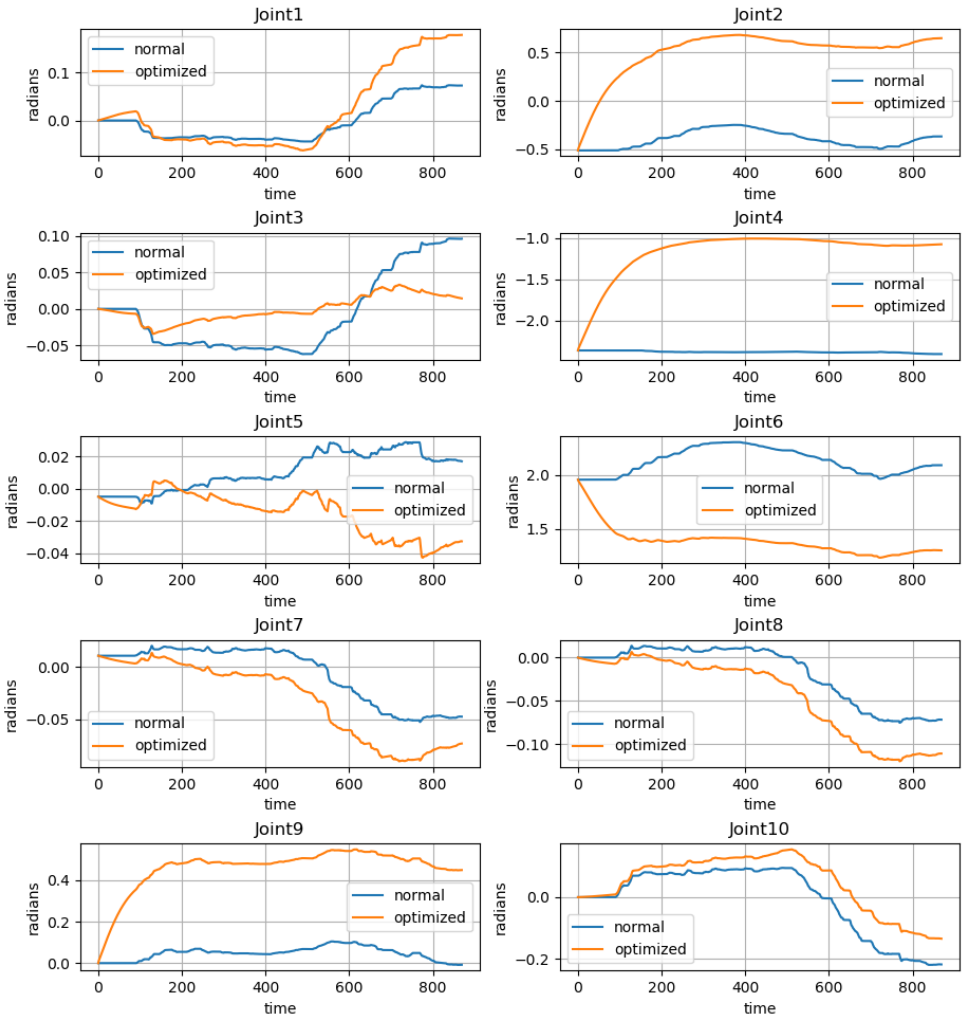


Figure 27: The joint angle difference between conventional kinematics and manipulability maximization during one experiment

Experiment Number	Conventional Kinematics	Manipulability Maximization
	Average Manipulability	Average Manipulability
1	0.3283	0.5855
2	0.3271	0.5837
3	0.3208	0.5657
4	0.3210	0.5622
5	0.3228	0.5767
6	0.3195	0.5639
7	0.3222	0.5614
8	0.3282	0.5841
9	0.3227	0.5669
10	0.3249	0.5757
Overall	**0.3237**	**0.5726**

Table 4: Average manipulabilities for each experiment

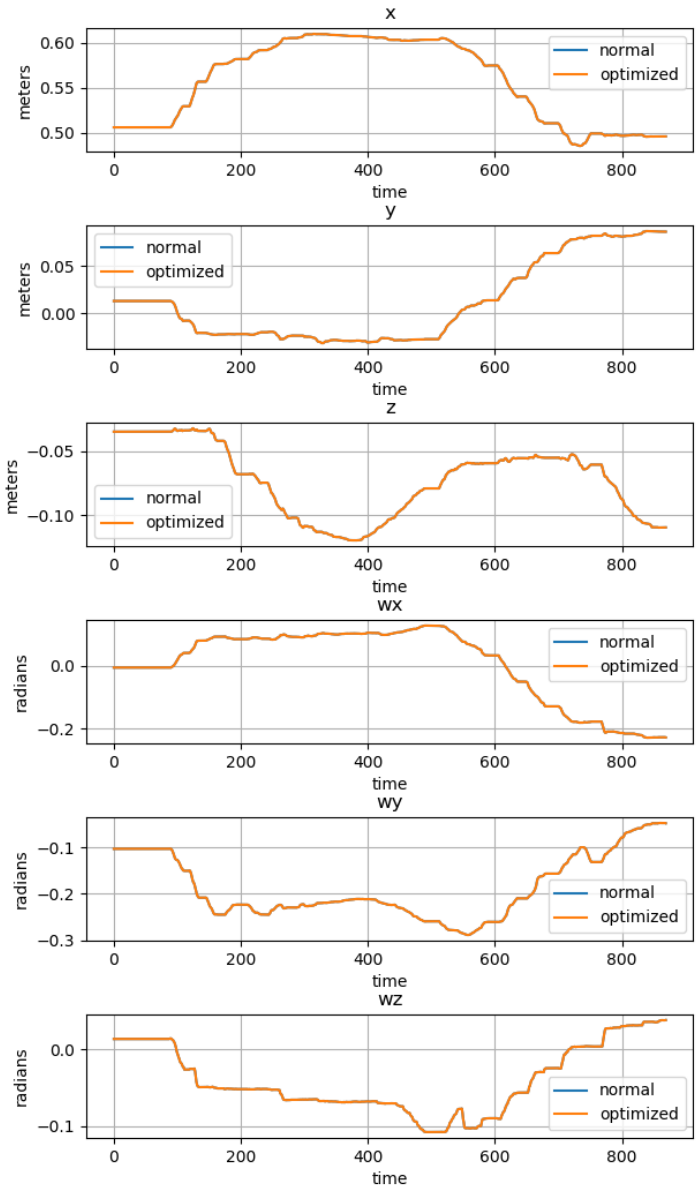


Figure 28: The end-effector pose difference between conventional kinematics and manipulability maximization during one experiment

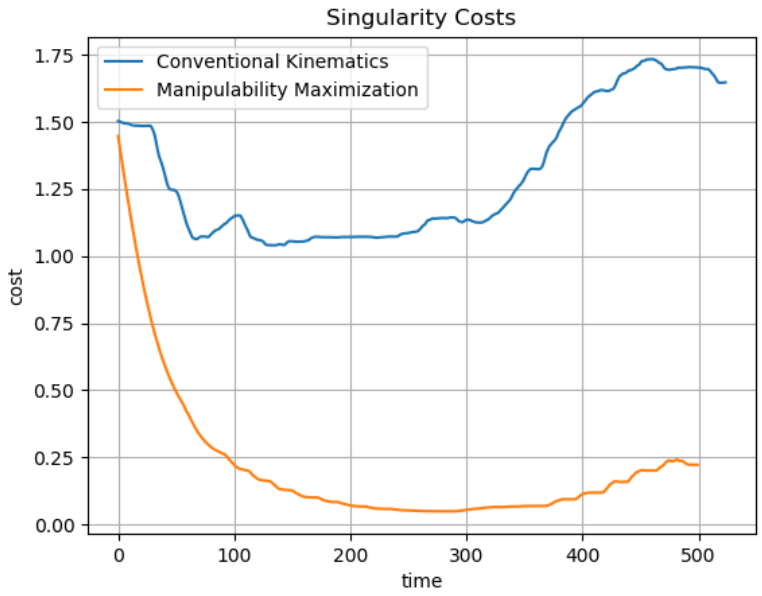


Figure 29: Singularity cost differences in the haptic cue generator between conventional kinematics and manipulability maximization

collected data is then compared with the previous haptic force cue experiments where the same amount of forces were applied. Figures 29, 30, and 31 shows the difference in the singularity, joint and overall costs respectively between the two kinematic systems. The average costs of each experiment for each kinematic system are also listed on table 5.

As theorized prior to the experiments, the singularity costs when manipulability maximization is used, is a lot less than when conventional kinematics is used. The amount of optimization on the singularity costs are also observed on the overall cost differences between the two systems. However, a higher manipulability does not necessarily mean a better joint configuration of the manipulator. In the joint costs, it is inspected that the conventional kinematics performs better. Also, in practice, because of such high joint costs in some of the configurations, overwhelming amounts of forces were experienced. Therefore, even though the overall costs of the haptic force cue generation with manipulability maximization was less than with the conventional kinematics, the experiment is not regarded as 100% successful.

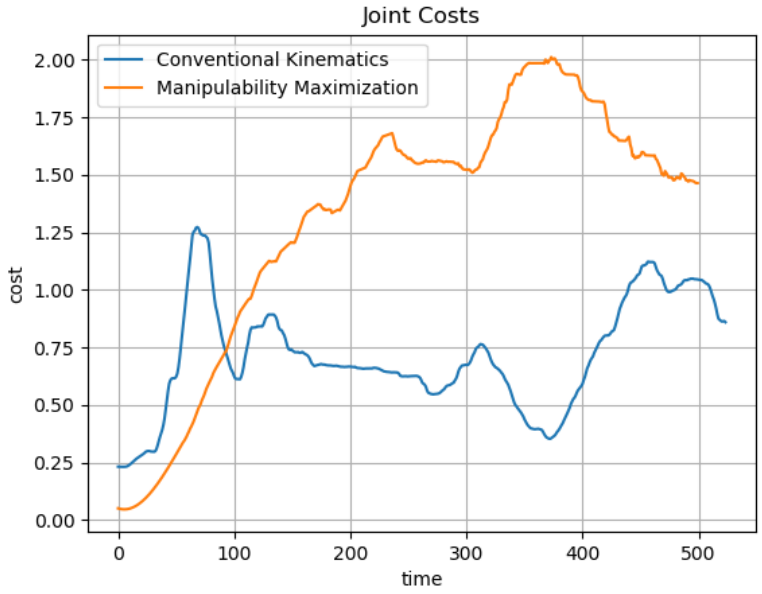


Figure 30: Joint cost differences in the haptic cue generator between conventional kinematics and manipulability maximization

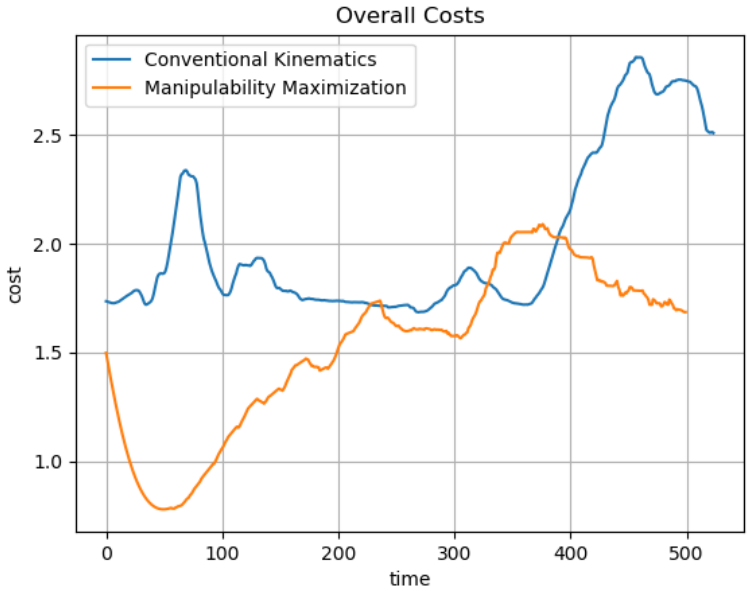


Figure 31: Overall cost differences in the haptic cue generator between conventional kinematics and manipulability maximization

Experiment Number	Haptic Force Cues with Conventional Kinematics			Haptic Force Cues with Manipulability Maximization		
	Average Joint Costs	**Average Singularity Costs**	**Average Overall Costs**	**Average Joint Costs**	**Average Singularity Costs**	**Average Overall Costs**
1	0.6508	0.3760	2.0268	0.8914	0.2567	1.1481
2	0.7129	1.3008	2.0137	0.6420	0.1914	0.8334
3	0.5939	1.3675	1.9614	0.9377	0.2319	1.1696
4	0.6435	1.3250	1.9686	1.3598	0.2136	1.5734
5	0.6750	1.3395	2.0144	1.0825	0.2100	1.2925
6	0.5849	1.3307	1.9156	0.9366	0.2542	1.1908
7	0.7135	1.3234	2.0370	0.9630	0.2552	1.2182
8	0.6071	1.3827	1.9897	1.2279	0.2195	1.4475
9	0.6602	1.3779	2.0380	0.9373	0.2369	1.1742
10	0.6134	1.3477	1.9612	0.9265	0.2398	1.1662
Overall	**0.6455**	**1.3471**	**1.9926**	**0.9905**	**0.2309**	**1.2214**

Table 5: Average joint, singularity and overall costs that compares haptic force cues with conventional kinematics and manipulability maximization

CHAPTER V

AUTONOMOUS CONTROL OF THE MANIPULATOR

This chapter explains the objectives, and the studies made for the autonomous control of the manipulator for the suturing task. Section 5.1, discloses the suturing task and explains the concepts that are used. Section 5.2 provides the reader the process of data collection. The graphical user interface that was designed as a bridge between the user and the autonomous sysem is explained in section 5.3. Finally, section 5.4, explains the designed feedback loop for turning the autonomous system into a closed loop system.

5.1 Suturing Task

Even though, a full autonomous surgery cannot be performed using the today's technology yet, some of the surgical subtasks can be transfered to the robotic systems for automation. Suturing task is a good candidate for automation because of the repetitiveness of its nature. For this work, an LfD (learning from demonstration) framework is used for automization of the suturing task. The localization of the instruments and environment (i.e. the needle, the thread and the tissue) is practiced by various imagind tecniques.

In order to simplify a complex trajectory, the suturing task is segmented into motion primitives. Therefore, the LfD models can be trained individually for each primitive and the generated trajectories can be replayed on the robotic system one at a time. The segmentations of the motion primitives and their intermediate steps are listed as:

- Suturing tasks begins.
- The robotic systems are brought the their home joint configurations.
- The initial position of the surgical needle is known and sen to the robotic

system. The primary robotic arm is moved towards the needle.

- The primary robotic arm grasps the surgical needle.
- The user selects their desired needle insertion point from the graphical user interface.
- The needle is inserted to the tissue from the selected positon.
- Tip of the needle's position and orientation are localized using imaging techniques for the secondary robotic arm to pull the needle out of the tissue.
- The secondary arm pulls the needle out.
- The needle is transfered back to the primary robotic arm.
- The thread of the needle is detected by the imaging techniques and a knot is tied using both robotic arms.
- Suturing task is finalized.

The final state of this project is planned to involve image proccessing techniques for real-time tracking of the surgical needle, the needle thread and the artificial tissue, a learning from demonstration method to generate the trajectories for the motion primitives, and a robotic system control scheme that will combine and execute these autonomous system on the physical instruments. The image processing techniques and machine learning methods are not part in the scope of this thesis. They are only briefly mentioned for the better grasping on automation of the suturing task.

5.2 Data Collection

For the learning from demonstration framework, a conventional tele-operation system has been prepared with a manipulator, haptic device, clutching pedals to collect data about the manipulator's status during the human demonstrations of the motion primitives. An operation table has been placed in front of the manipulator as a platform for the artificial tissue and the needle holder stand. The data collection setup has been shown in figure 32. The tele-operator picks the surgical needle up from a stand that holds the needle for easier grasping. Then it

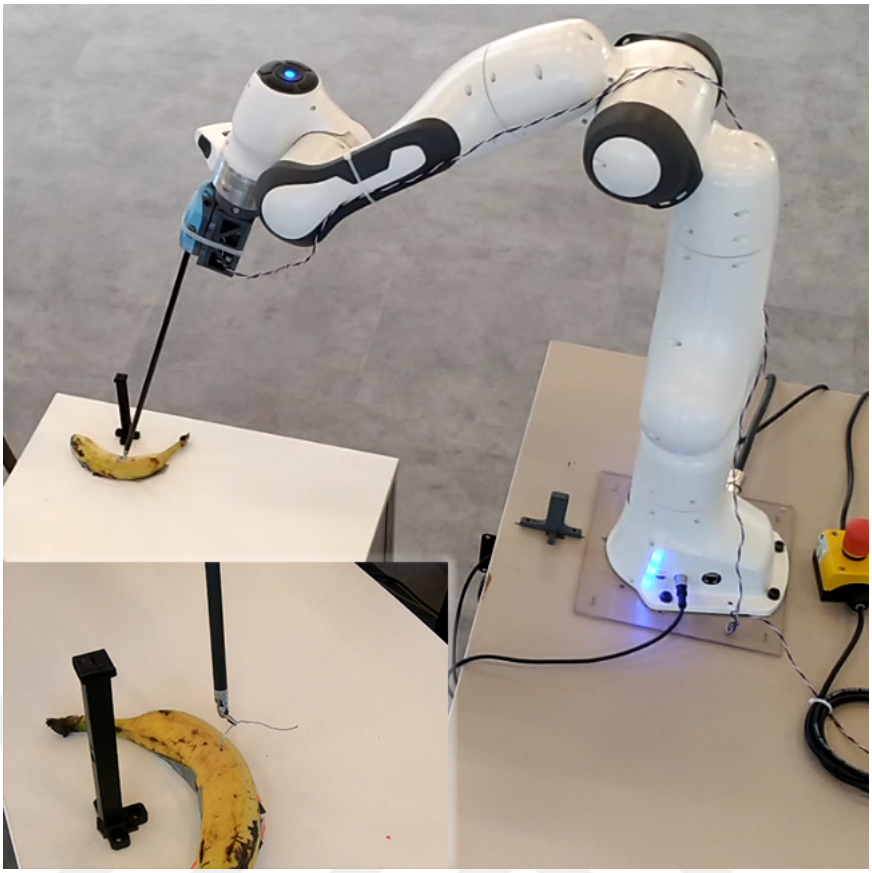


Figure 32: Data collection setup for LfD model

moves towards the artifical tissue to perform the suturing task. The status of the serial manipulator is saved during the whole operation.

Multiple information about the robotic system's current status is combined as a data frame and saved at each time step along with the current operational time. These information are:

- The end-effector pose of the serial manipulator (i.e. 7-DOF system)
- The end-effector pose of the whole robotic system (i.e. 10-DOF system)
- Joint torques
- Joint angles
- Joint velocities
- Estimated external forces
- External force sensor data

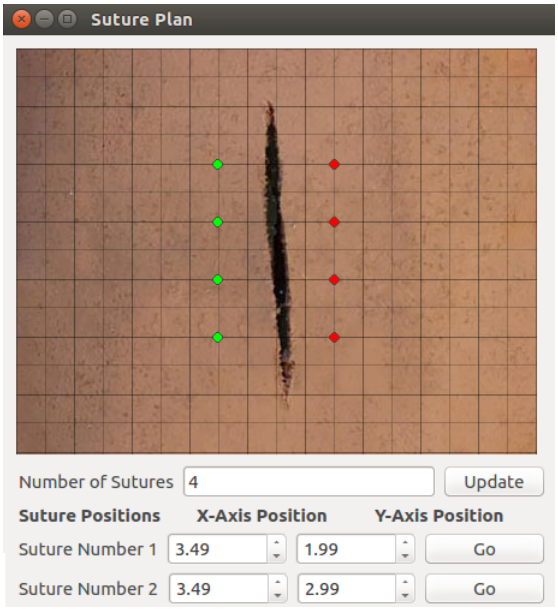


Figure 33: Suture plan graphical user interface designed as a bridge between the user and the autonomous control scheme

5.3 Graphical User Interface as a Bridge

In order to get insertion point inputs from the user, an adaptive graphical user interface (GUI) has been designed and demonstrated on figure 33. GUI is composed of two panels: one for visualizing the placements of the insertion points on an image of the artifical tissue, and another for changing the number of sutures and the positions of the insertion points. The image of the artifical tissue is marked with 0.5 cm apart grid lines to provide the user an intuitive guide for editing the insertion and exit points. During the initialization of the GUI, by default 10 insertion points are created and placed on the image equidistant from each other. Then, the user can change the number of sutures and the positions of insertion and exit points as they desire.

On the current implementation of the GUI design, the target tissue is assumed to be flat with a fixed vertical position. Therefore, the position that is picked from the 2D surface can easily be transformed to a 3D position and sent to the trained LfD models.

If the user edits the number of sutures or the positions of the insertion points, the changes are visualized on the first panel with the image of the artificial tissue in

real-time. After the user is satisfied with the number of sutures and the positions of the insertion and exit points, they can press "*Go*" in order to generate input positions for trained LfD models. The models generate trajectories for the motion primitive that they were trained for. Then, these trajectories are sent to the robotic system control schemes that were explained in chapter 3 and executed on the physical system.

5.4 Kinematic Feedback Loop

For an insertion operation to be successful, the needle has to be grasped from the correct position and the post-grasp orientation of the needle should be suitable and ready for insertion. In order to ensure a proper pose of the needle, the open loop structure of the control system should be turned into a closed loop one. Thus a kinematic-based feedback loop system has been designed to correct any errors that might occur on the trajectory during needle pick-up.

Since the initial position of the needle is previously known and fixed, two reference points are set for opening and closing the end-effector of the surgical tool. As long as the needle grasping trajectory is performed, the possible errors on the end-effector are calculated and corrected in the kinematic feedback loop, depending on the distance between the current configuration of the manipulator and the pre-determined reference positions. If the grasper has not yet been opened, the reference point for opening the grasper gets activated. If the point for opening the grasper has already been reached, the reference point for closing the grasper gets activated.

$\vec{q}_{final}$ angles in (32) are found by combining the feedforward angles that are generated by the trained LfD model ($\vec{q}_{LfD}$) and the feedback angles that are the pre-determined reference angles, with the ω parameter. The ω parameter has been used as a weighting parameter to determine which of the angle sets should be more dominant and their levels of dominances at the given time. The calculation of the ω parameter is shown in (33), where k is a scaling constant and the parameter d

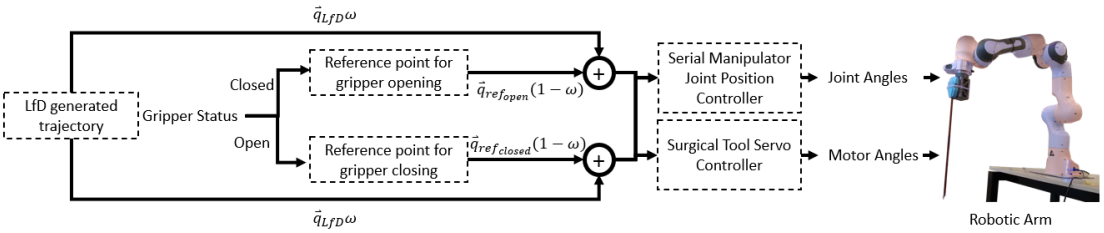


Figure 34: The kinematic feedback loop designed for turning the autonomous system into a closed loop system

is the Euclidean distance from the current angles to the active reference angles. The resulting ω parameter is always between 0 and 1. Having an ω value close to zero means the end-effector of the manipulator is close to the reference angles, thus the reference angles should be more dominant in $\vec{q}_{final}$. Having an ω value close to one means that the end-effector of the manipulator is far away from the reference angles, thus the model generated trajectories should be executed with little to no interference.

$$\vec{q}_{final} = \omega\ \vec{q}_{LfD} + (1-\omega)\ \vec{q}_{ref} \tag{32}$$

$$\omega = \text{-}1 + \frac{2}{1 + \exp(\text{-}k\exp(d))} \tag{33}$$

Figure 34, shows the dataflow of the closed loop autonomous control scheme with the kinematic-based feedback loop. The model generated trajectories are combined with either of the two pre-determined reference angles, using (32).

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

This chapter concludes the studies that were presented in this thesis and discusses the future works.

6.1 Conclusions

This thesis presented a tele-operation setup for suturing task, with various tele-operation optimization methods and the comparisons of them. It also demonstrated systems that were implemented for autonomous executions of the suturing task. As previously mentioned, a robot assisted surgical operation is more beneficial in terms of reducing the surgeon fatigue, hospitalization times and complications that may occur after the operation. The success chances of robot-assisted surgeries can be further improved by optimizing the tele-operations of the robotic surgical systems and automating some of the surgical sub-tasks. Therefore, various tele-operation optimization methods that would help the surgeon during the surgical tasks are implemented and compared. The previous studies in the literature that are related to to the subjects of this thesis were presented in chapter 2.

In chapter 3, the system overview, a generalized instrument setup and the control schemes of the different hardwares that were used throughout the studies were introduced. 2 different serial manipulators and 2 different haptic devices were used in different studies. The control schemes of all of these devices were explained. Furthermore, the decoupling of the surgical tool was also presented.

Chapter 4 first introduced the conventional kinematics calculations and defined the modelling of the robotic arms and the technical terms that were used in later sections. Section 4.2, proposed an adaptive inverse kinematics method where certain three joints of a 9-DOF redundant robotic arm is locked so that the system behaves like an actuated one. The method aimed to scale the velocity of the

end-effector by locking different sets of joints at a time according to the velocity input that the user gives to the kinematics. Therefore, the tele-operator can both make fine and precise movements, and fast movements without losing time. An experiment with 10 inexperienced subjects were conducted to comprehend the efficacy of the method. The results of the experiments showed that both the number of clutches and the duration of the experiment were reduced when adaptive inverse kinematics method was used compared to the traditional methods.

Section 4.3 presented an approach that would warn the user via force cues through the haptic device in case of a non-optimal pose of the manipulator's end-effector. The tele-operator may not realize when a joint is close to its limit or when the robotic arm is close to a singular pose. Therefore, directional haptic forces were generated to help the user to get out of the non-optimal pose and sent to the haptic device to warn them. Section 4.4 proposed a passive manipulability index maximization method to utilize the internal joint motions of the redundant robotic arm without moving its end-effector. Therefore, the tele-operation of the manipulator became more efficient without interfering the tele-operator's intended trajectory. The verifications and comparisons of the methods that were proposed in sections 4.3 and 4.4 were made with six different experimental setups explained in section 4.5.

Chapter 5 of the thesis presented the tools that help with the autonomous control of the suturing task. First, it briefly introduced the automation of suturing task. Then, it explained the tool that was implemented for data collection for the LfD framework training. Next, it talks about the graphical user interface that acts as a bridge between the user and the trained model to generate trajectories for the robotic system using user specified inputs. Finally, it proposed a kinematic feedback loop that turns the open loop control system into a closed loop one.

6.2 Future Work

The studies are mostly implemented and tested on a single-arm system even though suturing task requires a dual-arm system to be able to complete all the motion primitives (see section 5.1). The systems for the single-arm can easily be implemented on the dual arm system thanks to the modularity of the controller schemes. However, there should be an obstacle avoidance system to allow two manipulators to share a workspace. On top of that, section 4.3 can also generate force cues for the robotic arms to avoid each other. As the literature review suggests, the equation 27 in section 4.4 can be used for the internal motions of the manipulators to avoid colliding with each other.

The experiments in section 4.5 are simple manipulation tasks that might not do justice for the results of the proposed methods. Moreover, all 60 of the data for data anaalysis and verifications were collected from one person that would create a bias to the results. Even though, it is not healthy to collect results from six different setup from one subject, a human experiment with the best performing experimental setups can be designed to produce more accurate results and comparisons.

Bibliography

[1] P. C. Giulianotti, "Robotics in general surgery," *Archives of Surgery*, vol. 138, p. 777, July 2003.

[2] G. Martel and R. P. Boushey, "Laparoscopic colon surgery: Past, present and future," *Surgical Clinics of North America*, vol. 86, no. 4, pp. 867 – 897, 2006. Recent Advances in the Management of Benign and Malignant Colorectal Diseases.

[3] V. Vitiello, S. Lee, T. P. Cundy, and G. Yang, "Emerging robotic platforms for minimally invasive surgery," *IEEE Reviews in Biomedical Engineering*, vol. 6, pp. 111–126, 2013.

[4] J. Mahler, S. Krishnan, M. Laskey, S. Sen, A. Murali, B. Kehoe, S. Patil, J. Wang, M. Franklin, P. Abbeel, and K. Goldberg, "Learning accurate kinematic control of cable-driven surgical robots using data cleaning and gaussian process regression," in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 532–539, 2014.

[5] O. Ersoy, M. C. Yildirim, A. Ahmad, O. D. Yirmibesoglu, N. Koroglu, and O. Bebek, "Design and kinematics of a 5-dof parallel robot for beating heart surgery," in *2019 IEEE 4th International Conference on Advanced Robotics and Mechatronics (ICARM)*, pp. 274–279, 2019.

[6] O. Ersoy, "Development and control of a parallel endoscopic surgical robot platform for beating heart procedures," MS Thesis, Ozyegin University, Istanbul, June 2021.

[7] A. Kapoor, M. Li, and R. H. Taylor, "Spatial motion constraints for robot assisted suturing using virtual fixtures," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2005* (J. S. Duncan and G. Gerig, eds.), (Berlin, Heidelberg), pp. 89–96, Springer Berlin Heidelberg, 2005.

[8] G. Gras, K. Leibrandt, P. Wisanuvej, P. Giataganas, C. A. Seneci, M. Ye, J. Shang, and G. Yang, "Implicit gaze-assisted adaptive motion scaling for highly articulated instrument manipulation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4233–4239, May 2017.

[9] S.-Y. Oh, D. Orin, and M. Bach, "An inverse kinematic solution for kinematically redundant robot manipulators," *Journal of Robotic Systems*, vol. 1, no. 3, pp. 235–249, 1984.

[10] M. Vande Weghe, D. Ferguson, and S. S. Srinivasa, "Randomized path planning for redundant manipulators without inverse kinematics," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pp. 477–482, Nov 2007.

[11] Y. Ping, S. Hanxu, and J. Qingxuan, "Study on kinematics optimization of redundant manipulators," in *2006 IEEE Conference on Robotics, Automation and Mechatronics*, pp. 1–6, June 2006.

[12] D. Tolani and N. I. Badler, "Real-time inverse kinematics of the human arm," *Presence (Camb)*, vol. 5, no. 4, pp. 393–401, 1996.

[13] Y. Chen, Y. Chen, and J. Guo, "An optimization algorithm to expand the reduced workspace with fixed-joint method for redundant manipulator," in *2016 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*, pp. 116–120, July 2016.

[14] I. Zaplana and L. Basanez, "A novel closed-form solution for the inverse kinematics of redundant manipulators through workspace analysis," *Mechanism and Machine Theory*, vol. 121, pp. 829–843, March 2018.

[15] Z. Xu, S. Li, X. Zhou, W. Yan, T. Cheng, and D. Huang, "Dynamic neural networks based kinematic control for redundant manipulators with model uncertainties," *Neurocomputing*, vol. 329, pp. 255–266, 2019.

[16] M. Selvaggio, A. Ghalamzan Esfahani, R. Moccia, F. Ficuciello, B. Siciliano, *et al.*, "Haptic-guided shared control for needle grasping optimization in minimally invasive robotic surgery," in *IEEE/RSJ International Conference Intelligent Robotic System*, 2019.

[17] R. Pocius, N. Zamani, H. Culbertson, and S. Nikolaidis, "Communicating robot goals via haptic feedback in manipulation tasks," in *Companion of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 591–593, 2020.

[18] S. Parsa, D. Kamale, S. Mghames, K. Nazari, T. Pardi, A. R. Srinivasan, G. Neumann, M. Hanheide, and G. E. Amir, "Haptic-guided shared control grasping: collision-free manipulation," in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pp. 1552–1557, 2020.

[19] E. A. M. Ghalamzan, F. Abi-Farraj, P. R. Giordano, and R. Stolkin, "Human-in-the-loop optimisation: mixed initiative grasping for optimally facilitating post-grasp manipulative actions," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3386–3393, IEEE, 2017.

[20] M. Selvaggio, F. Abi-Farraj, C. Pacchierotti, P. R. Giordano, and B. Siciliano, "Haptic-based shared-control methods for a dual-arm system," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4249–4256, 2018.

[21] L. Marchal-Crespo, M. Bannwart, R. Riener, and H. Vallery, "The effect of haptic guidance on learning a hybrid rhythmic-discrete motor task," *IEEE Transactions on Haptics*, vol. 8, no. 2, pp. 222–234, 2015.

[22] F. Chinello, C. Pacchierotti, J. Bimbo, N. G. Tsagarakis, and D. Prattichizzo, "Design and evaluation of a wearable skin stretch device for haptic guidance," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 524–531, 2018.

[23] L. Meli, C. Pacchierotti, and D. Prattichizzo, "Experimental evaluation of magnified haptic feedback for robot-assisted needle insertion and palpation," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 13, no. 4, p. e1809, 2017.

[24] J. Bimbo, C. Pacchierotti, M. Aggravi, N. Tsagarakis, and D. Prattichizzo, "Teleoperation in cluttered environments using wearable haptic feedback," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3401–3408, 2017.

[25] H. Shen, H. Wu, B. Chen, Y. Jiang, and C. Yan, "Obstacle avoidance algorithm for 7-dof redundant anthropomorphic arm," *Journal of Control Science and Engineering*, vol. 2015, pp. 1–9, 01 2015.

[26] M. Dede, O. Maaroof, and E. Tatlicioglu, "A new objective function for obstacle avoidance by redundant service robot arms," *International Journal of Advanced Robotic Systems*, vol. 13, p. 1, 03 2016.

[27] L. Zhu, H. Mao, X. Luo, and J. Xiao, "Determining null-space motion to satisfy both task constraints and obstacle avoidance," in *2016 IEEE International Symposium on Assembly and Manufacturing (ISAM)*, pp. 112–119, 2016.

[28] K. Dufour and W. Suleiman, "On maximizing manipulability index while solving a kinematics task," *Journal of Intelligent & Robotic Systems*, pp. 1–11, 2020.

[29] Y. Zhang, Y. Huang, B. Cai, D. Guo, and Z. Ke, "Quadratic-programming based self-motion planning with no target-configuration assigned for planar robot arms," in *IEEE ICCA 2010*, pp. 534–539, 2010.

[30] Y. Zhang, D. Guo, K. Li, and J. Li, "Manipulability-maximizing self-motion planning and control of redundant manipulators with experimental validation," in *2012 IEEE International Conference on Mechatronics and Automation*, pp. 1829–1834, 2012.

[31] R. Jackson and M. Cavuşoğlu, "Needle path planning for autonomous robotic surgical suturing," *IEEE International Conference on Robotics and Automation : ICRA : [proceedings] IEEE International Conference on Robotics and Automation*, vol. 2013, pp. 1669–1675, 12 2013.

[32] T. Liu and M. C. Cavusoglu, "Needle grasp and entry port selection for automatic execution of suturing tasks in robotic minimally invasive surgery," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 552–563, 2016.

[33] B. Thananjeyan, A. Tanwani, J. Ji, D. Fer, V. Patel, S. Krishnan, and K. Goldberg, "Optimizing robot-assisted surgery suture plans to avoid joint limits and singularities," in *2019 International Symposium on Medical Robotics (ISMR)*, pp. 1–7, 2019.

[34] H. Kang and J. Wen, "Autonomous suturing using minimally invasive surgical robots," in *Proceedings of the 2000. IEEE International Conference on Control Applications. Conference Proceedings (Cat. No.00CH37162)*, pp. 742–747, 2000.

[35] C. Staub, T. Osa, A. Knoll, and R. Bauernschmitt, "Automation of tissue piercing using circular needles and vision guidance for computer aided laparoscopic surgery," in *2010 IEEE International Conference on Robotics and Automation*, pp. 4585–4590, 2010.

[36] S. Iyer, T. Looi, and J. Drake, "A single arm, single camera system for automated suturing," in *2013 IEEE International Conference on Robotics and Automation*, pp. 239–244, 2013.

[37] S. Sen, A. Garg, D. V. Gealy, S. McKinley, Y. Jen, and K. Goldberg, "Automating multi-throw multilateral surgical suturing with a mechanical needle guide and sequential convex optimization," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4178–4185, 2016.

[38] M. Kam, H. Saeidi, S. Wei, J. Opfermann, S. Leonard, M. Hsieh, J. Kang, and A. Krieger, *Semi-autonomous Robotic Anastomoses of Vaginal Cuffs Using Marker Enhanced 3D Imaging and Path Planning*, vol. 11768, pp. 65–73. Springer International Publishing, 10 2019.

[39] S. A. Pedram, P. Ferguson, J. Ma, E. P. Dutson, and J. Rosen, "Optimal needle diameter, shape, and path in autonomous suturing," *CoRR*, vol. abs/1901.04588, 2019.

[40] Y. Tian, M. Draelos, G. Tang, R. Qian, A. N. Kuo, J. A. Izatt, and K. Hauser, "Toward autonomous robotic micro-suturing using optical coherence tomography calibration and path planning," *CoRR*, vol. abs/2002.00530, 2020.

VITA

Begüm Sunal received her B.Sc. degree in the Department of Computer Science at Özyeğin University, Ístanbul, Turkey in 2018. Then, she started her M.Sc. studies in Computer Science at Robotics Laboratory of Özyeğin University, under the supervision of Dr. Özkan Bebek and Prof. Dr. Erhan Öztop. Her main research interests are robotics and control systems.