NTNU – Trondheim
Norwegian University of
Science and Technology

Teleoperation of Mobile Robot Manipulators

Bjørn Heber Skumsnes

Master of Science in Engineering Cybernetics
Submission date: June 2012
Supervisor: Geir Mathisen, ITK
Co-supervisor: Aksel A. Transeth, SINTEF Anvendt Kybernetikk

Norwegian University of Science and Technology
Department of Engineering Cybernetics

NTNU
Norges teknisk-naturvitenskapelige universitet

Fakultet for informasjonsteknologi, matematikk og elektroteknikk
Institutt for teknisk kybernetikk

MASTER THESIS

Kandidatens navn: **Bjørn Heber Skumsnes**

Fag: **Engineering cybernetics**

Oppgavens tittel (norsk): **Teleoperering av mobile robotmanipulatorer**

Oppgavens tittel (engelsk): **Teleoperation of mobile robot manipulators**

Bakgrunn:

SINTEF and NTNU are investigating how to remotely control operations on offshore oil platforms by employing robots. A lab facility has been established to investigate such operations, and the lab is equipped with a Phantom OMNI "joystick" which can provide force feedback to an operator. The joystick can be used to control a Kuka robot manipulator. This is known as bilateral control in the field of telerobotics.

The field of telerobotics is perhaps one of the earliest aspects of robotics. Applications are plentiful and range from space robotics to medical systems and rehabilitation.

The main focus of this project will be to do a literature survey of telerobotic systems and control architectures, and to develop a framework for further development together with a teleoperation system based on the methodology found from the survey.

Tasks

- Perform a literature study on teleoperation of mobile manipulators
- Design a control system for teleoperation of the Seekur Jr + Schunk LWA3 mobile manipulators based on the literature study.
    - Develop a teleoperation system for 6 DOF control (position + orientation) of the robot end-effector.
    - Account for time-varying communication delays.
    - Present force-feedback to operator in a "good" manner (e.g. limited vibrations in haptic joystick).
    - Investigate how to keep the manipulatbility of the robot arm high during teleoperation control.
    - Account for joint limitations.
- Implement the teleoperation control system with an available robot simulator in ROS and evaluate the results.

Oppgaven gitt: 09.01.2012
Besvarelsen leveres: 04.06.2012
Besvarelsen levert:
Utført ved Institutt for teknisk kybernetikk
Veileder: Forsker Aksel A. Transeth, SINTEF Anvendt Kybernetikk

Trondheim, den 09.01.2012

Geir Mathisen
Faglærer

Abstract

With ever cheaper and more versatile robots, the use of robotic systems increases rapidly. Although robots are becoming more intelligent, the cognitive capabilities of humans can still not be matched. By combining the intelligence of a human operator with the strength, endurance and size of a robot, in addition to separating the robot and operator to avoid danger to the operator, the applications are innumerable. The use of an operator to remotely control a robot is often referred to as teleoperation.

In a teleoperation system it is important to present the state of the robot and the remote environment with high accuracy and in a comprehensible way. With a large number of sensor data, a solution is to enhance the feeling of telepresence or transparency of the system. That is, making the human operator feel like he or she is interacting directly with the manipulated environment. This could be achieved by using a haptic joystick, which is able to generate force feedback to the operator, to present information about the slave robot. Examples of such informations are the distance to an obstacle or deviation from a desired movement. Such a system is often called a bilateral teleoperation system, where the stability is especially sensitive to transmission delay. This time-delay is often introduced by the communication network between the human operator and remote robot.

This thesis presents a control architecture for interpreting a change in the joystick position to a desired end-effector velocity for a mobile manipulator. In addition to calculating the velocity, the controller is designed to comply with the joint limits, optimize the manipulability and handle time-varying transmission delay. A force, that depends on the deviation between the desired and measured end-effector position, is sent back to the human operator, as well as a visual feedback. To increase the precision of the end-effector movement the position of the movable robot base is fixed when the manipulability is above a given threshold, and moves only to increase the workspace of the robot. The designed system is implemented using Robot Operating System (ROS) and tested on a virtual mobile manipulator. The virtual robot

is based on a model of a Schunk LWA3 7-DOF manipulator, mounted on a Seekur Jr. wheeled mobile base.

Several experiments prove that the system with the proposed control architecture is stable when under influence of constant, as well as variable time-delay. Any standard deviation between the measured and desired end-effector position is eliminated, and the trajectory of the end-effector is almost identical the desired, though delayed when affected by communication delay. Neither the force feedback nor end-effector position show indications of dramatic change at the transition between fixed and moving robot base. Simulations with human operators show that they are able to move the end-effector of a virtual mobile manipulator from an initial position to a predefined goal, with the use of a Phantom Omni, haptic joystick.

Sammendrag

Med stadig billigere og mer allsidige roboter, øker bruken av robotsystemer raskt. Selv om roboter blir stadig mer intelligente, kan menneskets kognitive evner fortsatt ikke sammenlignes. Ved å kombinere intelligensen til en menneskelig operatør med styrken, utholdenheten og størrelse til en robot, i tillegg til å skille roboten og operatøren for å unngå fare for operatøren, er bruksområdene utallige. Det å bruke en operatør til å fjernstyre en robot blir ofte referert til som teleoperering.

I et teleoperert system er det viktig å presentere tilstanden til roboten og det eksterne miljøet med høy nøyaktighet og på en forståelig måte. Med mye måledata, er en løsning å øke følelsen av telesamvær eller gjennomsiktighet i systemet. Det vil si, å få den menneskelige operatøren til å føle at han eller hun er i direkte samhandling med det manipulerte miljøet. Dette kan oppnås ved å bruke en haptisk joystick, som er i stand til å generere tilbakemelding i form av kraft til operatøren, til å presentere informasjon om slaveroboten. Eksempler på slike opplysninger er avstanden til en hindring eller avvik fra en ønsket bevegelse. Et slikt system kalles ofte et bilateralt, teleoperert system, hvor stabiliteten er spesielt følsom for forsinkelse i overføringen. Denne tidsforsinkelsen er ofte introdusert av kommunikasjonsnettverk mellom den menneskelige operatøren og eksterne roboten.

Denne avhandlingen presenterer en kontrollarkitektur for å tolke en endring i joystick-ens posisjon til en ønsket hastighet for ytterpunktet til en mobil manipulator. I tillegg til å beregne hastigheten, er kontrolleren designet for å overholde begrensningene for leddene, optimalisere manipulerbarheten og håndtere tidsvarierende overføringsforsinkelse. En kraft, som avhenger av avviket mellom ønsket og målt ytterpunktsposisjon, sendes tilbake til den menneskelige operatøren, i tillegg til en visuell tilbakemelding. For å øke presisjonen på bevegelsen til ytterpunktet er posisjonen til den bevegelige basen låst når manipulerbarheten er over en gitt terskel, og beveger seg bare for å øke arbeidsområdet til roboten. Det utviklet systemet er implementert ved hjelp av Robot Operating System (ROS) og testet på en virtuell mobil ma-

nipulator. Den virtuelle roboten er basert på en modell av en Schunk LWA3 7-DOF manipulator, montert på en Seekur Jr. mobil base med hjul.

Flere forsøk viser at systemet med den foreslåtte kontrollarkitekturen er stabilt både når det er påvirket av konstant, så vel som variabel tidsforsinkelse. Eventulle standardavvik mellom målt og ønsket posisjon for ytterpunktet går mot null, og banen til ytterpunktet er nesten identisk med den ønskede, men forsinket når systemet er påvirket av kommunikasjonsforsinkelse. Verken den taktile tilbakemeldingen eller posisjonen til ytterpunktet viser tegn til dramatisk endring rundt overgangen mellom fast og bevegelig robotbase. Tester med menneskelige operatører viser at de klarer å flytte ytterpunktet til en virtuell mobil manipulator fra en gitt utgangsposisjon til et forhåndsdefinert mål, ved å bruke en Phantom Omni, haptisk joystick.

Preface

This thesis is based on the work done in the final semester of the Master of Science Degree program in Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU). Much of the knowledge prior to this study is based on the research from the project, written in the fall 2011, under the same supervisors.

I would like to thank Geir Mathisen from department of engineering cybernetics for accepting me as a student and supervising this thesis. A sincere thanks is expressed to Aksel Andreas Transeth from SINTEF Applied Cybernetics for help and guidance throughout the project regarding the problem description and reviewing of my thesis.

Contents

Chapter 1

Introduction

1.1 Motivation

Teleoperation, which literally means *operating from a distance*[1], is often used similar to *remote control* and describes systems where a human operator and a robot that interacts with an environment, are parted by a distance. The cognitive capabilities of a human operator combined with the strength, endurance and size of a robot makes teleoperation a versatile tool [2, 3], used in innumerable applications. The ability to control robots from a remote place will further extend the use of robots.

A human operator is, in most cases, more capable to handle unexpected scenarios, as opposed to an autonomous robot. However, in many cases it would be preferable to have the human operator in another location than the robot when conducting a task. Such tasks could be operations in dangerous environment, limited area of movement, or in a remote area for a longer period. With ever cheaper and more diverse robots, an increasing number of tasks that former have been human labor, can now be done by robots.

Examples of task dangerous for a human operator are operations in war zones [4, 5], such as mine fields [6, 7] and explosive removal [8], handling of

[1]The prefix *tele* comes from Greek and means "at a distance" [1]

hazardous materials, such as in nuclear power plants [2, 9], mining [10, 11], and for search and rescue operations [12]. Tasks where the human operator would use a lot of effort to reach the area of operation includes underwater exploration [13] and inspection [14], in-space robotics [15], offshore robotics [16] and surveillance [17]. Telerobotics are also beneficial when the area of operation is limited or the desired movement is very small or large, such as in medical surgery [18] or for large space manipulators [19].

Teleoperation is one of the earliest areas of robotics [20] and modern teleoperation systems are based on more than hundred years of research and study. However, since the communication takes place over a network that may introduce a time delay, teleoperation is also one of the most challenging areas of robotics [20]. The next section presents a short historical overview of the evolution of bilateral teleoperation and what areas of research the scientists have focused on.

1.2 Background

This section gives a short description of the terms, theory and historical background needed to understand the challenges regarding teleoperation and meaning of the contribution from this study. More background theory is presented in Chapter 2, including a more detailed description of a bilateral teleoperation system. Different controllers proposed to stabilize systems with time delay, and properties of mobile manipulators are the main areas of interest.

As opposed to the term *remote control*, teleoperation is usually used for systems where the human operator is provided with a sense of telepresence. Telepresence refers to making a person feel like he or she is present or has an effect, at a place other than their true location. This means making the difference between an interaction with the manipulated and the technical mediated environment as small as possible. One way of enhancing the telepresence is by using bilateral teleoperation, in addition to multimodal

feedback [21]. Bilateral teleoperation is used to describe systems where the human operator interacts with a manipulator and experiences similar forces as the manipulator interacting with the environment. Multimodal feedback refers to the use of multiple modes of feedback to present the information to the human operator, such as visual, tactile or acoustic feedback. Figure 1.2 shows bilateral teleoperation with multimodal feedback.

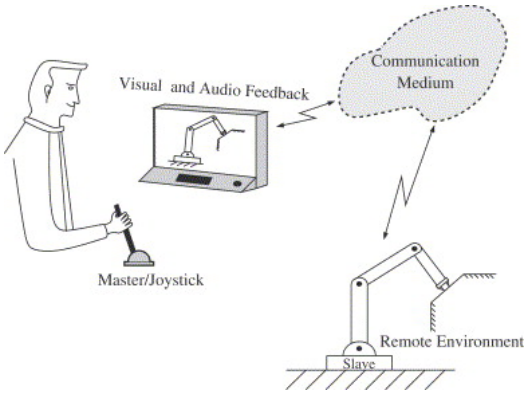


Figure 1.1: Bilateral teleoperation with multimodal feedback [22]

While the history of teleoperation started with Nikola Tesla in 1898, when he demonstrated the first radio-controlled vessel [23], the history of bilateral teleoperation is said to have started with Raymond C. Goertz and his first work in the mid 1940s [24]. Since Goertz built the first master-slave teleoperation system, the focus of the research has shifted through the time.

The first area of interest was the effect of time delay in the communication network, where several experiments were conducted by Sheridan and Ferrell (1963) [25] and Ferrell (1965) [26]. In Ferrell (1966) [27], force feedback was tested under the effect of time delay, and stability became one of the main areas of research. To address the problem of delays, Ferrell and Sheridan (1967) [28] developed an approach called supervisory control. Here, the human operator gives high-level directives to the robot and receives summary information in return [29].

In the late 1980s and early 1990s, the focus shifted towards network theory through impedance representation [30], hybrid representation [31], scattering

theory [32] and passivity based control [33]. The intension was to model the communication network and include the transmission delay in the stability analysis. Later, the researchers addressed the degree of transparency [34, 35], that is, how well the feedback to the operator reflects the state of the robot.

After the introduction of the Internet, the topic of research shifted towards problems arising when using packet switched networks [36, 37]. In modern time, the researchers have sought to develop more intelligent and adaptive control schemes [38, 39, 40], as well as using teleoperation on new applications. One of the more recent areas of use is mobile robots [41, 42, 43].

Along with a robot interacting with the environment, an important part of a bilateral teleoperation system is the human operated joystick. As a part of the multimodal feedback, tactile (or force) feedback is frequently used and provided through the joystick. To get this effect, it is necessary to use a haptic joystick, which can send force back to the human operator while measuring the force and position applied by the operator. To connect a haptic joystick to a computer without interacting with the hardware device, it is necessary to use an *application programming interface* (API). The APIs differ in what programming language they are written in, what devices they support, and whether they are open source or only for commercial use.

A more detailed presentation of the background theory is covered in the next chapter and is used as a basis for this study. The contribution of this work is the next topic.

1.3 Contribution

In the early stage of this study, a comprehensive literature survey was performed. An overview of the most important systems in teleoperation is given, with a selection of them described in more detail. The intension with the survey is to acquire knowledge about bilateral teleoperation of mobile manipulators and to investigate similar applications and proposed solutions, as well as giving the reader enough background theory in order to understand the rest of the work done in this master thesis. The main focus of the survey was mobile manipulators, operational space control, output synchronization of bilateral teleoperation systems, and possible combinations of the aforementioned topics, as well as any practical implementations.

Based on the literature survey, a control architecture for teleoperation of a mobile manipulator is proposed. The mobile manipulator consists of a Schunk LWA3 (a 7-link robot arm) mounted on a Seekur Jr. (a four wheeled mobile base). The control scheme includes an interpretation of the position and orientation of a Phantom Omni haptic joystick, with a total of 6 degrees of freedom (DOF), in addition to a calculation of a force feedback in 3 DOF.

The control architecture is stable under the influence of variable time-delay and solves a set of predefined tasks, where the tasks are arranged in a hierarchy. With descending priority, the tasks are given as: comply with joint limits, move the end-effector according to the movement of the human operated joystick, and optimize the manipulability. From the these tasks, a desired velocity for each of the manipulator joints and mobile base is calculated.

A force feedback is designed based on the end-effector position for the mobile manipulator and the joystick position. The force is sent to the human operator through the haptic joystick and depends on the position deviation and linear joystick velocity, where the joystick velocity is averaged to give a smooth feedback.

Though a physical mobile manipulator exists, the designed controller is tested on a personal computer using a physics-engine-based simulator. The simulator is used to calculate, and graphically represent the motion of a virtual mobile manipulator based on the desired velocities.

The control scheme is implemented using an application programming interface (API) named Robot Operating System (ROS) [44], where the program is written in Python and C++ programming language. In addition to a set of convenient libraries, ROS provides an interface between the developed program and joystick and between the program and simulator.

Since the master and slave manipulator are kinematically different, such as different numbers of joints, it is necessary to use operational space to compare the movements of the slave end-effector and the joystick. As a result, the forward, forward differential and inverse differential kinematics are calculated for the mobile manipulator, while only the forward kinematics for the orientation of the joystick are found. The reason for omitting the kinematics for the joystick position is that this is included in the API. Nor is it interesting to find the inverse kinematics for the orientation, since the force feedback is not given in this dimension for the joystick.

The designed framework, used to implement the controller proposed in this study, is based on a framework provided by SINTEF. The provided framework consists of a haptic joystick and a physics-engine based simulator with the model for the mobile manipulator.

The framework, as well as the control architecture, is tested in a variety of scenarios to investigate the stability and handling of time delay. In addition, analyses on how intuitive and informative the operation of the system is, are carried out. The latter properties include the magnitude of the force feedback and how the movement of the human operated joystick is interpreted to a desired movement of the mobile manipulator.

The rest of this thesis describes the work done in this study in further detail, where the next section presents the outline of this thesis.

1.4 Outline

The remaining part of this report is categorized into eight additional chapters, organized as follows.

In Chapter 2, a more detailed presentation of the background material is given. This chapter covers most of the theory from a literature survey done in the early stage of the study.

The transitions between operational and joint space are described using kinematics, where the derivation of the kinematics for the human operated joystick and mobile manipulator is presented in Chapter 3.

In Chapter 4 the proposed control architecture is introduced. The architecture calculates the desired joint velocities for the mobile manipulator and a force feedback to the human operator. The calculations are based on the configuration of both the mobile manipulator and joystick.

A framework is created for implementation of the control architecture. An overview of the system is covered in Chapter 5.

The most important parts of the implemented program are described in Chapter 6. The program includes the control architecture and communication with the joystick and the physics-engine-based simulator.

Chapter 7 introduces the experimental setup and several experiments. A quantitative and qualitative analysis of the developed teleoperation system is given, before the results from these tests are presented.

The test results are compared with the desired behavior and discussed in Chapter 8. The discussion covers the design choices for the framework and control architecture, where the measured properties include the degree of stability and position tracking and how intuitive, informative and predictable the operation is.

Chapter 9 concludes the thesis, discusses areas of improvements and presents several recommendations for further work.

Chapter 2

Background

This chapter summarizes some of the articles read and studies done in the early stage of this thesis. The intention of this literature study is to get an idea of what scientists have done of research and to understand the concept behind teleoperation. The knowledge acquired here is used later to derive methods for solving the different problems.

First in this chapter, Section 2.1 and Section 2.2 cover two terminologies, important in the design of the control architecture used in this study. The robot controlled in this thesis is a mobile manipulator, and Section 2.3 describes the different aspects of mobile manipulators. A short overview of different components in a bilateral teleoperation system is presented in Section 2.4, which is partially based on the project report by Skumsnes (2011) [45]. Section 2.5 covers some of the different control architectures proposed for teleoperation systems, with much of the theory also based on Skumsnes (2011) [45]. In Section 2.6 a short description of the program used for implementation is given, before the last section summarizes the chapter.

2.1 Equations of Motion in Operational Space

The standard way of writing the equations of motion for a robot system is to use generalized, or joint, coordinates, $\mathbf{q} \in \mathbb{R}^n$. If the relationship between the generalized coordinate, and the position and orientation of the end-effector, described in global, Cartesian coordinates, $\mathbf{x} \in \mathbb{R}^m$, is known, it is possible to describe the equations of motion in operational space.

When analyzing the movement in joint space, it is necessary to define either a desired position, velocity or acceleration for each joint. In most cases, the precise value of a joint is irrelevant, while the end-effector position, velocity, acceleration or applied force is of more interest and more intuitive. Instead of defining a desired end-effector position, it is common to define a desired end-effector velocity, or acceleration based on the deviation in the position.

First in this section, the derivation of the operational space formulation for a system is described, before presenting the procedure for creating a hierarchical control, based on the equations of motion in operational space.

2.1.1 Operational Space Formulation

The operational space formulation is a way of describing the equations of motion by using the end-effector location $\mathbf{x}$, given in Cartesian coordinates. The location, $\mathbf{x}$, could either include orientation of the end-effector, or just the position, depending on the application. In the most general case, the location is described by six variables, three position and three orientation variables. For a basic system, each of the entries of the generalized torque, $\tau \in \mathbb{R}^n$, corresponds to a torque or force at the corresponding joint. By using the operational space formulation, the generalized torque can be partitioned into one part that ensures motion of the end-effector, and one part controlling the contact force at the end-effector [46].

One way of finding the equations of motion in operational space, is to start with the equations of motion in joint space, the most common way of de-

scribing the dynamics of a manipulator. The reason for using joint space is that the relationship between the force acting on the end-effector, and the velocity and acceleration of the end-effector depend on the configuration of the entire manipulator, not only the location of the end-effector.

The equations of motion in joint space can be written as

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \tau \tag{2.1}$$

where $\mathbf{D}(\mathbf{q})$ is the joint inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ the Coriolis and centrifugal effects, $\mathbf{g}(\mathbf{q})$ the gravity torque vector, $\mathbf{q} \in \mathbb{R}^n$ the generalized coordinates, and τ the set of joint torques. Khatib (1987) [46] suggested to use the Jacobian matrix, $\mathbf{J}(\mathbf{q})$, defined as

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{2.2}$$

to describe the end-effector equations of motion in operational space. The acceleration in operational space is found as the derivative of the velocity (2.2), written as

$$\ddot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{J}}(\mathbf{q})\dot{\mathbf{q}}. \tag{2.3}$$

By substituting the expression for the joint acceleration, $\ddot{\mathbf{q}}$, found from the equations of motion (2.1), into the expression for the acceleration in operational space (2.2), the following relationships can be defined

$$\begin{aligned} \mathbf{\Lambda}(\mathbf{q}) &= [\mathbf{J}(\mathbf{q})\mathbf{D}^{-1}(\mathbf{q})\mathbf{J}^T(\mathbf{q})]^{-1} \\ \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}) &= [\tilde{\mathbf{J}}^T(\mathbf{q})\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{\Lambda}(\mathbf{q})\dot{\mathbf{J}}(\mathbf{q})]\dot{\mathbf{q}} \\ \eta(\mathbf{q}) &= \tilde{\mathbf{J}}^T(\mathbf{q})\mathbf{g}(\mathbf{q}) \\ \mathbf{F}_x &= \tilde{\mathbf{J}}^T(\mathbf{q})\tau. \end{aligned} \tag{2.4}$$

Here $\mathbf{\Lambda}$ is the operational space inertial matrix, $\mathbf{\Gamma}$ the centrifugal and Coriolis forces in operational space, η the gravity forces, and $\mathbf{F}_x$ the forces acting on

the end-effector. With

$$\tilde{\mathbf{J}}(\mathbf{q}) = \mathbf{D}^{-1}(\mathbf{q})\mathbf{J}^{T}(\mathbf{q})\mathbf{\Lambda}(\mathbf{q}) \tag{2.5}$$

the end-effector equations of motion in operational space can be written [47] as

$$\mathbf{\Lambda}(\mathbf{q})\ddot{\mathbf{x}} + \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}) + \eta(\mathbf{q}) = \mathbf{F}_x. \tag{2.6}$$

The variables in (2.4) are used for a redundant system, where the Jacobian is not invertible. For a non-redundant system the variables can be simplified to

$$\begin{aligned} \mathbf{\Lambda}(\mathbf{q}) &= \mathbf{J}^{-T}(\mathbf{q})\mathbf{D}(\mathbf{q})\mathbf{J}^{-1}(\mathbf{q}) \\ \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}) &= [\mathbf{J}^{-T}(\mathbf{q})\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{\Lambda}(\mathbf{q})\dot{\mathbf{J}}(\mathbf{q})]\dot{\mathbf{q}} \\ \eta(\mathbf{q}) &= \mathbf{J}^{-T}(\mathbf{q})\mathbf{g}(\mathbf{q}) \\ \mathbf{F}_x &= \mathbf{J}^{-T}(\mathbf{q})\tau. \end{aligned} \tag{2.7}$$

The system (2.6) is subject to the operational force $\mathbf{F}_x$ if and only if the manipulator, described by (2.1), is controlled by the following generalized joint force vector [46]

$$\tau = \mathbf{J}^{T}(\mathbf{q})\mathbf{F}_x + \mathbf{N}^{T}(\mathbf{q})\tau_0, \tag{2.8}$$

where $\mathbf{I}_n$ is the $n \times n$ identity matrix, τ_0 is an arbitrary joint force vector, and $\mathbf{N}(\mathbf{q}) = \mathbf{I}_n - \tilde{\mathbf{J}}(\mathbf{q})\mathbf{J}(\mathbf{q})$ is the null-space.

The following operational force is proposed by Khatib (1987) [46]:

$$\mathbf{F}_x = \mathbf{F}_c^* + \mathbf{\Lambda}(\mathbf{q})\mathbf{F}_m^* + \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}) + \eta(\mathbf{q}), \tag{2.9}$$

where $\mathbf{F}_m^*$ is the decoupled end-effector command vector, and $\mathbf{F}_c^*$ the contact forces acting at the end-effector. $\ddot{\mathbf{x}}$ can be replaced by $\mathbf{F}_m^*$ because $\ddot{\mathbf{x}}$ is acting on a system that acts like a unit point mass. By selecting the following frame,

the forces and motion can be controlled in the operational frame

$$\mathbf{F}_x = \mathbf{F}_m + \mathbf{F}_c, \tag{2.10}$$

where

$$\begin{aligned} \mathbf{F}_m &= \mathbf{\Lambda}(\mathbf{q})\Omega\mathbf{F}_m^* + \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}) + \eta(\mathbf{q}) \\ \mathbf{F}_c &= \bar{\Omega}\mathbf{F}_c^* + \mathbf{\Lambda}(\mathbf{q})\bar{\Omega}\mathbf{F}_s^*. \end{aligned} \tag{2.11}$$

Here Ω, $\bar{\Omega}$ and $\mathbf{F}_s^*$ represent the complementary task specification matrices, and the vector of end-effector velocity damping in the direction of force control, respectively.

2.1.2 Hierarchical Control

Hierarchical control is used to describe control architectures where different tasks are divided into a hierarchical structure. These tasks often include end-effector position, manipulator pose, joint limitation, and obstacles avoidance.

For a redundant manipulator, where the number of generalized coordinates, n, exceeds the number of parameters needed to describe the position and orientation of the end-effector, m, several combination of joint values may give the same position for the end-effector. This makes it possible to partition the torque further, into a part which ensures that additional tasks are met.

Nakanishi et al. (2008) [48] propose to exploit the redundancy by designing the desired joint velocity, $\dot{\mathbf{q}}_d \in \mathbb{R}^n$, as follows

$$\dot{\mathbf{q}}_d = \mathbf{J}^\dagger \dot{\mathbf{x}}_d + \left(\mathbf{I}_n - \mathbf{J}^\dagger \mathbf{J}\right) \mathbf{w}_1, \tag{2.12}$$

where $\dot{\mathbf{x}}_d \in \mathbb{R}^m$ is the desired end-effector velocity and $\mathbf{w}_1$ is an arbitrary vector.

The desired joint acceleration, $\ddot{\mathbf{q}}_d$, is found from the expression for the acceleration in operational space (2.3), that is

$$\ddot{\mathbf{q}}_d = \mathbf{J}^{\dagger}(\ddot{\mathbf{x}}_d - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \left(\mathbf{I}_n - \mathbf{J}^{\dagger}\mathbf{J}\right)\mathbf{w}_2, \tag{2.13}$$

where $\ddot{\mathbf{x}}_d \in \mathbb{R}^m$ is the desired end-effector acceleration, $\dot{\mathbf{J}}$ is the time derivative of the Jacobian, and $\mathbf{w}_2$ is an arbitrary vector, similar to $\mathbf{w}_1$. By finding the analytic time derivative of (2.12), and compare the result with (2.13), De Luca et al. (1992) [49] show that the vector $\mathbf{w}_2$ can be written as

$$\mathbf{w}_2 = \dot{\mathbf{J}}^{\dagger}\mathbf{J}(\dot{\mathbf{q}} - \mathbf{w}_1)\dot{\mathbf{w}}_1. \tag{2.14}$$

The vectors $\mathbf{w}_1$ and $\mathbf{w}_2$ can be designed to minimize a function, for instance, a function describing the pose or use of energy, by either velocity or acceleration.

Sentis and Khatib (2005) [50] suggest to exploit the redundancy as well, but based on the expression for the generalized joint force vector (2.8). The main idea is to assign additional tasks in the null-space of the previous tasks, creating a hierarchy with several layers.

If the movement of the manipulator is unconstrained, the following control law for solving a predefined task, while ensuring a desired posture, is used

$$\tau = \tau_{task} + \tau_{posture}, \tag{2.15}$$

with

$$\tau_{task} = \mathbf{J}_t^T\mathbf{F}_t \tag{2.16}$$

$$\mathbf{F}_t = \mathbf{\Lambda}_t(\mathbf{q})\ddot{\mathbf{x}}_{t(ref)} + \mathbf{\Gamma}_t(\mathbf{q}, \dot{\mathbf{q}}) + \eta_t(\mathbf{q}), \tag{2.17}$$

where $\mathbf{x}_t(\mathbf{q}) \in \mathbb{R}^m$ is a vector of the coordinates for an operational task; $\ddot{\mathbf{x}}_{t(ref)}$ the reference input for the acceleration; $\mathbf{\Lambda}_t$, $\mathbf{\Gamma}_t$ and η_t are the mass matrix, Coriolis and centrifugal forces, and gravity forces, respectively, and are defined as (2.4). $\mathbf{F}_t$ is the force acting on the end-effector to solve the

specified task, while the Jacobian $\mathbf{J}_t$ is defined as

$$\mathbf{J}_t(\mathbf{q}) = \partial \mathbf{x}_t(\mathbf{q})/\partial \mathbf{q}. \tag{2.18}$$

A posture criteria is controlled by the control input, $\Gamma_{posture}$, with coordinates $\mathbf{x}_p(\mathbf{q})$ and Jacobian $\mathbf{J}_p(\mathbf{q}) = \partial \mathbf{x}_p(\mathbf{q})/\partial \mathbf{q}$. The control force is defined as

$$\tau_{posture} = \mathbf{J}_{p|t}^T \mathbf{F}_{p|t} \tag{2.19}$$

$$\mathbf{F}_{p|t} = \mathbf{\Lambda}_{p|t}(\mathbf{q})\ddot{\mathbf{x}}_{p(ref)} + \mathbf{\Gamma}_{p|t}(\mathbf{q}, \dot{\mathbf{q}}) + \eta_{p|t}(\mathbf{q}). \tag{2.20}$$

Here, $\mathbf{J}_{p|t} = \mathbf{J}_p \mathbf{N}_t$, $\ddot{\mathbf{x}}_{p(ref)}$ is a reference input for the posture, and $\mathbf{\Lambda}_{p|t}$, $\mathbf{\Gamma}_{p|t}$ and $\eta_{p|t}$ are defined as in (2.4), with $\mathbf{J} = \mathbf{J}_{p|t}$. The control law can now be written as

$$\tau = \tau_{task} + \mathbf{N}_t^T [\mathbf{J}_p^T \mathbf{F}_{p|t}]. \tag{2.21}$$

The previous control architecture can be further extended to handle constraints. By giving the handling of constraints the highest priority, the control law is given as

$$\tau = \mathbf{J}_{constraints}^T \mathbf{F}_{constraints} + \mathbf{N}_{constraints}^T \tau_{task}. \tag{2.22}$$

Assuming that maintaining a given posture is regarded as a low- priority task, a general, multi-level hierarchy, control law can be written as

$$\begin{aligned} \tau = \tau_{constraints} + \mathbf{N}_{constraints}^T (\tau_{task(1)} + \mathbf{N}_{task(1)}^T (\tau_{task(2)} \\ + \mathbf{N}_{task(2)}^T (\tau_{task(3)} + ... \mathbf{N}_{task(N-1)}^T \tau_{task(N)}))). \end{aligned} \tag{2.23}$$

By defining an extended null-space matrix,

$$\mathbf{N}_{prec(k)} = \mathbf{N}_{task(k-1)} \mathbf{N}_{task(k-2)} ... \mathbf{N}_{task(1)} \mathbf{N}_{constraints}, \tag{2.24}$$

the nested topology can be written as

$$\tau = \tau_{constraints} + \tau_{1|prec(1)} + \tau_{2|prec(2)} + ... + \tau_{N|prec(N)}, \tag{2.25}$$

where $\tau_{k|prec(k)} = \mathbf{N}^T_{prec(k)}\tau_{task(k)}$ are the prioritized controls. The subscript $k|prec(k)$ is used to indicate that the kth task is projected into the null-space of all preceding tasks and constraints. As a result, the task Jacobian is projected into the null-space $\mathbf{N}_{prec(k)}$

$$\mathbf{J}_{k|prec(k)} \triangleq \mathbf{J}_k\mathbf{N}_{prec(k)}. \tag{2.26}$$

The tasks $\mathbf{x}_k(\mathbf{q})$ are controlled within the hierarchy by choosing the following control torque:

$$\tau_{k|prec(k)} = \mathbf{J}^T_{k|prec(k)}\mathbf{F}_{k|prec(k)} \tag{2.27}$$

$$\mathbf{F}_{k|prec(k)} = \mathbf{\Lambda}^T_{k|prec(k)}\ddot{\mathbf{x}}_{k(ref)} + \mathbf{\Gamma}_{k|prec(k)} + \eta_{k|prec(k)}. \tag{2.28}$$

Here $\mathbf{\Lambda}_{k|prec(k)}$, $\mathbf{\Gamma}_{k|prec(k)}$ and $\eta_{k|prec(k)}$ are defined as in (2.4), with $\mathbf{J} = \mathbf{J}_{k|prec(k)}$.

The null-space of task k is the areas of motion with no force effects on the preceding levels in the hierarchy. This makes it possible to control the joint to first comply with the restrictions, then fulfill several tasks with descending priority. The tasks with lowest priority often cover poses that are more beneficial for manipulability and force effect.

Manipulability is a measurement of how easy the end-effector can move in different directions. The next section describes this term more thoroughly.

2.2 Manipulability

Depending on the configuration of a manipulator, the magnitude of the adjustment in joint configuration required for a small position change in a given direction for the end-effector varies. The size of the area where the end-effector can move with a small change in the joint configurations, is referred to as manipulability.

When the degree of freedom for the end-effector is two or larger, it is necessary

to have a way of comparing the manipulability in a single dimension.

2.2.1 Manipulability Measure

Consider a robot arm, where the configuration is given by the generalized coordinates $\mathbf{q}_a = [q_{a1} \quad q_{a2} \quad ... \quad q_{an}]^T$, the location of the end-effector by the operational coordinates $\xi_a = [\xi_{a1} \quad \xi_{a2} \quad ... \quad \xi_{am}]^T$, and the direct instantaneous kinematic model by

$$\dot{\xi}_a = \mathbf{J}_a(\mathbf{q}_a)\dot{\mathbf{q}}_a. \tag{2.29}$$

The subset of realizable operational velocities $\dot{\xi}_a$, such that the corresponding joint velocities satisfy $\|\dot{\mathbf{q}}_a\| \leq 1$, is an ellipsoid in the m-dimensional space containing $\dot{\xi}_a$ [51]. The algebraic measurement of this ellipsoid is often called manipulability measure.

Yoshikawa [52, 53] developed different ways of measuring the manipulability, including the more usual version

$$w = \sigma_{a1}\sigma_{a2}...\sigma_{am}, \qquad w \geq 0 \text{ (0 is worst)}, \tag{2.30}$$

which is proportional to the ellipsoid volume. σ_{ai} are the singular values from a Singular Value Decomposition (SVD) of the Jacobian, $\mathbf{J}_a$, and satisfy $\sigma_{a1} \geq \sigma_{a2} \geq ... \geq \sigma_{am}$. It can be shown that this corresponds to $w = \sqrt{\det(\mathbf{J}_a(\mathbf{q}_a)\mathbf{J}_a^T(\mathbf{q}_a))}$. Another, more qualitative, measure suggested, is the ratio of the minimum and maximum radii of the ellipsoid

$$w_2 = \frac{\sigma_{am}}{\sigma_{a1}}, \qquad 0 \text{ (worst)} \leq w_2 \leq 1 \text{ (best)}.$$

In addition to the two aforementioned measures, Yoshikawa defines two more measures, w_3 and w_4, that are further described in Yoshikawa (1985) [53].

2.2.2 Manipulability for Mobile Manipulators

Bayle, Fourquet and Renaud (2001) [54] defined a manipulability measure, w_5, extending the notion of eccentricity of the ellipse:

$$w_5 = \sqrt{1 - \frac{\sigma_m^2}{\sigma_1^2}}. \tag{2.31}$$

They showed that this measure can be used to describe the manipulability of mobile manipulators, using the singular values of $\bar{\mathbf{J}}$, defined as

$$\bar{\mathbf{J}}(\mathbf{q}) = \mathbf{J}(\mathbf{q}) \begin{bmatrix} S(\mathbf{q}_b) & 0 \\ 0 & \mathbf{I}_n \end{bmatrix}, \tag{2.32}$$

satisfying

$$\dot{\xi} = \bar{\mathbf{J}}(\mathbf{q})\mathbf{u}, \tag{2.33}$$

with $\mathbf{u} = [\mathbf{u}_b^T \ \dot{\mathbf{q}}_a^T]^T$ containing the velocities for the system , $\mathbf{q}_b$ describing the configuration of the mobile base, and the Jacobian $\mathbf{J}(\mathbf{q})$ satisfying $\dot{\xi} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}$. The matrix $S(\mathbf{q}_b)$ is given by the configuration instantaneous kinematic model [55] for a movable platform

$$\dot{\mathbf{q}}_b = S(\mathbf{q}_b)\mathbf{u}_b. \tag{2.34}$$

Here the operational coordinate ξ defines the configuration and location of the end-effector in global coordinates, while $\mathbf{u}_b$ represent the platform controls. A planar mobile manipulator, with configuration defined by $\mathbf{q} = [x \quad y \quad \phi \quad q_{a1} \quad q_{a2}]^T$, can be seen in Figure 2.1. The scale for the manipulability measure w_5, ranges from 1 (worst) to 0 (best), with the ellipsoid corresponding to $\|\mathbf{u}\| \leq 1$, instead of $\|\dot{\mathbf{q}}\| \leq 1$.

A manipulability measure can be used to design a controller for a mobile manipulator, to keep the instantaneous reachable area for the end-effector as large as possible.

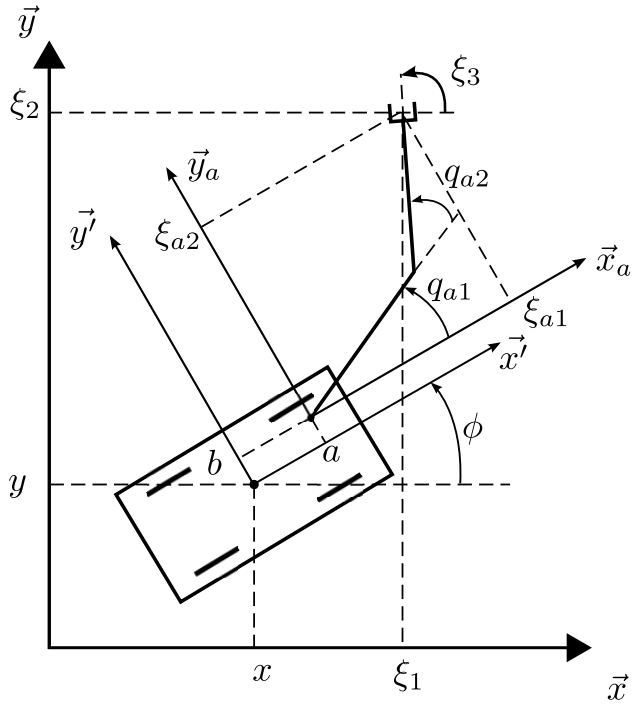


Figure 2.1: A planar mobile manipulator (n=2) [54, 56].

2.3 Mobile Manipulators

The expression "manipulator" is defined as a person or mechanical device that manipulates the environment [57], and is in robotics often used to refer to some sort of robotic arm. A manipulator usually consist of several links connected with motor-driven joints, where at least one link is connected to a fixed surface. The joints can either translate or rotate the links, with the overall goal to place the end-link, called the end-effector, to a given position with a desirable orientation. Manipulators have been used in the industries since George Devol designed the first programmable robot in the mid-1950s [58]. A modern manipulator is shown in Figure 2.2.

Mobile robots are usually used to describe robots that are not connected to one physical location, and have the ability to move around in their environment. The applications of mobile robots ranges from underwater and aerial vehicles, to land robots, and are used both by the industry, military

Figure 2.2: A modern manipulator [59].

and civilian consumers. The propulsion of land robots can be tracks, legs or wheels, which are most common. A popular mobile robot used in research is shown in Figure 2.3

Figure 2.3: Allegedly the world's most popular mobile robot for research [60].

A mobile manipulator is often used to describe a multi-link manipulator mounted upon a mobile platform [61]. Figure 2.4 shows a typical mobile manipulator. The combination of a manipulator and mobile robot exploits the advantages from each systems, and reduces their drawbacks.

Figure 2.4: Mobile Manipulator [62].

2.3.1 Redundancy

The definition of redundancy varies between authors. McKerrow describes it in "Introduction to Robotics" [63] as redundancy, when a manipulator can reach a specified position with more than one configuration of linkages. In "Robot Control" [64], a robotic system is said to be redundant when the way of achieving a given task is not unique. According to Samson and Borgne (1990) [64], a system is called *truly redundant* when there exists an infinite set of solutions in the joint space, for a given end-effector configuration. In "Advanced Robotics" [65], *kinematic redundancy* is used to describe a system where it is possible to change the internal structure of configuration of the mechanisms, without changing the position or orientation of the end-effector.

When a manipulator is mounted on a mobile base, the total number of degrees of freedom (DOF) usually increases. If the total DOF exceeds the DOF

needed to solve a task, the system could become redundant. If the manipulator is redundant initially, the mobile base could increase the redundancy, making the system kinematic redundant, and/or truly redundant. The increase of DOF can also expand the set of configurations that can provide a given end-effector position and orientation. This means that a given task can be solved in several, and in some cases infinite, number of ways.

The introduction of redundancy complicates the calculation and the derivation of a controller, especially when the system is controlled by a human operator, and the input has less DOF than the mobile manipulator. For example, what happens when the operator pushes a joystick forward? Should the base, manipulator or both move? It also requires a more intelligent controller to decide the best joint configuration for each task.

If an obstacle prevents the system from solving a task with a given configuration, a redundant system may solve the same task with another configuration. A redundant system may also maintain a more beneficial posture, in terms of mobility, force capability and stability. With an outstretched manipulator, the directions where the end-effector can move without moving the base becomes limited. The mobile manipulator is also more unstable when the center of gravity is placed outside the mobile base. Some posture can even reduce the load on joints with restriction on the applied torque.

2.3.2 Nonholonomic Constraints

One of the major challenges with mounting a manipulator on a mobile base is the potential introduction of nonholonomic constraints. A nonholonomic system is often used to describe a system whose state depends on the path taken to achieve it, meaning that one or more constraints are not integrable. A constraint is said to be nonholonomic if it is a constraint on the velocity, limiting the directions of the movement.

Usually, the mobile base consists of two or four wheels or tracks, and the instantaneous direction of movement is limited to forward and backward.

Figure 2.5: URANUS omni-directional mobile robot [66].

One way of solving this problem is the use of omni-directional wheels, shown in Figure 2.5, making it possible to travel in all directions. If omni-directional wheels or equivalent solutions are not used, a typical constraint for a mobile base is

$$\dot{x}\sin\phi - \dot{y}\cos\phi = 0, \tag{2.35}$$

where x, y is the position of the base, and ϕ the angle between the x-axis and the direction of movement. Figure 2.6 shows a mobile robot with non-holonomic constraints.

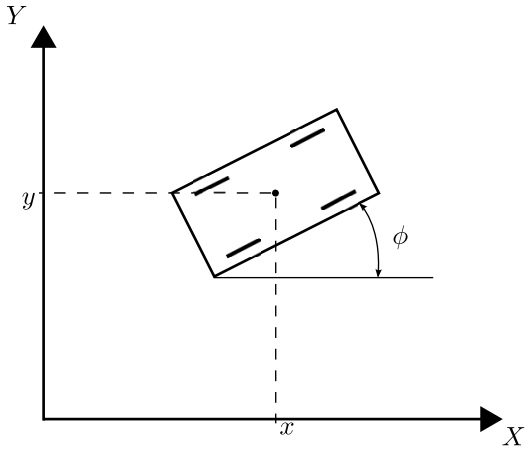


Figure 2.6: The robot can only move in the direction of the wheels and is therefore affected by nonholonomic constraints.

2.3.3 Increased Workspace

One of the main advantages with mobile manipulators, is the extension of the area of influence for the end-effector, called workspace. A mobile manipulator has the same reachable area as an infinite numbers of regular manipulators along the path where the mobile base can move.

As a result of the increased workspace, a mobile manipulator can replace one or several manipulators with fixed base. In addition, a set of mobile manipulators can eliminate the need for a moving production line, which could be both difficult and expensive to implement when the objects get very large. This means that the use of mobile manipulators in the industry is potentially much cheaper than the use of regular manipulators.

If the mobile manipulator is controlled by a human operated manipulator, or joystick, over a transmission line with time delay, it is said to be part of a teleoperation system. The definition of such system, and the components within will be described in the next section.

2.4 Components of a Bilateral Teleoperation System

It is common to define a bilateral teleoperation system as a system consisting of five elements: a human operator, a master manipulator, a communication channel, a slave manipulator, and an environment. When the human operator is presented by a tactile, in addition to a visual, display, the system is said to be bilateral. The block diagram in Figure 2.7 is commonly used to represent a teleoperation system [32]. Depending on the authors of the different papers, the human operated joystick is often referred to as the *leader*, or *local* or *master* manipulator, and the robot that interacts with the environment as the *follower*, or *remote* or *slave* manipulator. In this thesis, master and slave will be used to refer to the human operated manipulator and the manipulator interacting with the environment, respectively.

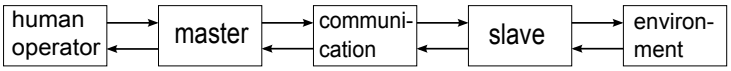


Figure 2.7: Block diagram of a teleoperation system [32]

The rest of this section is based on Figure 2.7, and each of the block will be investigated. Except the last subsection, this section is mostly based on the work done in Skumnses (2011) [45], with some changes for better understanding. The first link of a teleoperation system is the human operator. To make any stability analysis applicable, it is necessary to assume that the operator is stable. The human operator communicates with the rest of the teleoperation system through a master manipulator, which in this case is a haptic device. The next step, covered by Section 2.4.1, is therefore to describe a haptic device. The calculation is often done on a computer, and various interfaces between the computer and a haptic device, often referred to as application programming interface, are discussed in Section 2.4.2. The information calculated on the local computer is sent over a communication channel, which may introduce a time delay. Section 2.4.3 will cover some of the problems generated. The last subsection describes some applications for teleoperation systems, in other words, different slave manipulators.

2.4.1 Haptic Devices

A haptic device, often some sort of joystick, is a device that supports force feedback to the operator. This feedback is usually based on movement and reaction of a controlled object. The device can apply force, vibration, and/or motion to the user. This mechanical stimulation may be used in the creation and control of virtual objects, and for improvement of remote control of machines and devices (telerobotics).

Figure 2.8 shows a Phantom Omni, a haptic joystick with 3 DOF force feedback and 6 DOF positional sensing. A more extensive survey, including the investigation of different haptic devices, can be found in Skumsnes (2011) [45].

Figure 2.8: A Phantom Omni.

To read the position of and send force back to the haptic joystick, it is necessary to have a way to communicate and control the joystick. This interface, called application programming interface, will be explained in the next subsection.

2.4.2 Application Programming Interface

An application programming interface (API) serves as an interface between the haptic device and the computer. There are different methods of implementing haptic device control into an application, ranging from the lowest driver layer, to the highest scene graph layer. The last section covers different APIs available for the general user.

Some of this subsection is based on a shortened version of the work done in Skumsnes (2011) [45], in addition to a new API.

Driver Layer

The driver layer provides the fastest and the most precise response, but it demands a great effort to get the device working. Support of any other device that does not have compatible communication protocol means rewriting lot of source code. Optimized and well documented drivers written in C or C++

programming language are often provided by the manufacturers of haptic devices.

Low-level API

A low-level API hides the kinematics algorithm implementation from the programmer, and allows developers to work directly with position, rotation, and force vectors in the application. Many low-level APIs work as a common interface for different drivers, which is very helpful when supporting a lot of haptic devices. A low-level API is often a good choice when good haptic performance is needed while using own graphic rendering method. Haptic rendering is a haptic interaction, processing in virtual scene, where a convincing force reaction at the edge of a complex object is created [67].

High-level scene API

A high-level API often includes low-level APIs for hatics, graphics, physics, and audio processing. The objects in the virtual world are usually organized in a tree structure, with a specific root node, such as a world node. It is possible to apply graphical and haptical properties to an object, and set the specific properties recursively to its children object. A high-level haptic API is often the best choice for prototyping an application when the speed of development is crucial and performance is not a priority.

Specific APIs

Several APIs are discussed in Skumsnes (2011) [45]. An alternative to the APIs mentioned here is Robot Operating System (ROS)[44], which is a software framework with standard operating system services. Drivers for haptic devices are not originally included in the program, but different research labs and projects have published several open source codes for such devices. A more detailed explanation of ROS is presented in Section 2.6.

The computer which the API is implemented on, is usually connected to the slave manipulator through a communication network. Telerobotics differ from ordinary robotics since the signals are sent over such communication network. This network can make the teleoperation system unstable, due to the time delay that may occur. The potential problem with time delay is therefore one of the main areas of research, and will be the next topic.

2.4.3 Time Delay in Teleoperation

One of the key characteristics of a teleoperation system is the introduction of time delay. The handling of time delay is therefore an importing part when designing a controller. Time delay occurs when the communication channel in Figure 2.7 is stretched over a great distance, or through a slow medium.

Even though the first study on time delay in teleoperation system appeared in 1963 [25], instability as a result of time delay was not a problem before 1966, when time delay was used in the presence of force feedback [27]. Time delay combined with force feedback can create force reflections, and destabilize a bilateral teleoperation system. Force reflection is used to describe the phenomenon when the reference given to the slave manipulator is affected by the force applied from the slave controller, two time delays earlier.

First in this subsection, two main types of time delay are discussed, before a way of limiting the problems with time delay is presented in 2.4.3.

Types of Delay

The time delays that occur can be divided into two categories: constant and variable time-delay. The reason why type is important, is that many of the stability analyses used for different control architectures are only applicable on system with constant time-delay.

Constant time-delay is usually a result of a stretched communication channel,

or communication through a slow but consistent medium. The size of this type of delay is predictable and time-invariant.

Variable time-delay occurs when the signals are sent over for example Internet, or other equivalent packet based communication net where individual packages can be lost, or to movable objects, where the distance between the senders and receivers varies. Furthermore, such a delay can occur when signals are sent through a variable medium, as through the ocean, air or space, where the properties of the medium and signal velocities change with time. Systems with variable time-delay are less predictable, and more difficult to keep stable.

Buffer

To avoid the problems with variable time-delay, Lee, Martinez-Palafox and Spong (2006) [68] propose to consider variable time-delay as constant time-delay, by using data-buffering. If the data arrives in correct order and stored in a buffer, it is possible to read the data with a constant rate, treating it as constant delay. This simplifies the calculation, but will slow the teleoperation system significantly if the variations are large, since the constant delay will be chosen equal to the highest assumed value of the delay.

This would, however, not solve the problem with packet loss.

2.4.4 Applications with Teleoperation Systems

The device which interacts with the environment is often called slave or remote manipulator, and is usually controlled by a local controller using input sent from the human operator. The variety of slave manipulators used in teleoperation is great, so is the area of use.

Handling Hazardous Material

One of the main benefits of using teleoperation system is the separation of the human operator and the manipulated environment, making it very desirable in handling hazardous materials.

Raymond C. Goertz started the research on teleoperation system in the mid 1940s, building the first master-slave teleoperator [22]. The first systems were controlled using an array of on-off switches to activate different motors and move various axis [69]. But according to Goertz, the lack of feel made the manipulators "slow and somewhat awkward to operate". In 1951 Goertz built the first teleoperation system with force feedback, using steel cables and pulleys [69, 70]. The design was used to handle radioactive material from behind a shielded wall, shown in Figure 2.4.4.

Figure 2.9: Raymond C. Goertz used bilateral teleoperation in early 1950s to handle radioactive material [58].

While nuclear applications was one of the main areas of research in the early history of teleoperation, the interest shifted to other areas in the 1980s and 1990s, when nuclear power activity began to decline. In modern times, the applications of teleoperation in hazardous environment include detecting leaks of sealed radioactive materials [71], disarming explosives, mainte-

nance of high-voltage electrical power lines, and search and rescue in disaster zones [72].

Underwater Vehicles

As the use of nuclear power declined, the interest for teleoperation for underwater vehicles grew, making unmanned underwater vehicles for scientific exploration, or military use, one of the main applications of teleoperation during the 1970s and 1980s. In Uhrich (1973) [73], one of the earlier control architectures for underwater manipulator with force feedback is presented.

Space Robotics

The main motivations for using teleoperation in space robotics are the reduced cost of assembly, maintenance and repair tasks in space and the increased safety for the astronauts.

Since the design by Goertz was based entirely on mechanical coupling between the master and slave arms, the range between the operator and the manipulated environment was limited. The first teleoperation system with force feedback, while separating master and slave electronics, was Central Research Laboratory model M2 of 1982, and was used to verify the assembly of space truss structures.

Bejczy et al. (1994) [74] developed a dual-arm bilateral teleoperation system at the Jet Propulsion Laboratory (JPL), for space applications. For the first time, the master and slave systems were kinematically and dynamically different, requiring control in Cartesian space coordinates. The system was used for simulating teleoperation in space.

There are numerous applications of use of bilateral teleoperation in space robotics [75, 76, 77, 78, 79, 80]. Experiments were conducted by Imaida et

al. (2004) [81] and Yoon et al. (2004) [82] on teleoperation of a robotic arm, with six DOF, on board the Engineering Test Satellite 7.

Telesurgery

There are several benefits using teleoperations in surgery, often referred to as telesurgery. Not only does it save time, money and effort by allowing medical expertise to be exchanged around the world without requiring the physician to travel, it can also reduce the trauma to the patient by allowing procedures to be performed through small incisions [83]. By scaling the movement, the surgeon can conduct operations with higher precision than traditional surgery.

Mobile Robots

A new application of bilateral teleoperation is mobile robots [84, 41, 85, 42]. Since real-time visual feedback from the mobile robot requires a high bandwith and the camera has a limited viewing angle, force feedback is used to give the human operator a good impression about the environment surrounding the mobile robot.

A special case of mobile robots is mobile manipulators, where a manipulator is mounted on a mobile base.

As mentioned earlier, instability can occur when time delay is introduced in combination with force feedback. The next section covers one of the most common controllers for bilateral teleoperation systems, used to ensure stability.

2.5 Controllers for Teleoperation Systems

This section presents a set of important tools used to design stable controllers for bilateral teleoperation systems, including the most commonly used controllers. In addition to the controllers described in this section, there exists a numerous variants and combinations of the same controllers. The reason for presenting these particular controllers, is to get an idea of the different approaches used to create a control scheme for a bilateral systems.

An important tool, used to prove and ensure stability of a controller, is scattering theory, and many controllers are designed either based on this theory directly or on the results from this theory. Scattering theory is described in Section 2.5.1, which is an extension of the work in Skumsnes (2011) [45], altered to improve the reader's understanding. In Section 2.5.2 different approaches for ensuring position and/or velocity tracking for identical master and slave manipulator, in addition to handle time delay, are discussed, also based on Skumsnes (2011) [45]. Section 2.5.3 covers some methods for modeling a virtual environment, before a variety of controllers proposed for mobile manipulators are presented in the last subsection.

2.5.1 Scattering Theory

One of the major breakthroughs in handling instability caused by time delay in presence of force feedback, was achieved by Anderson and Spong (1989a) [32]. The teleoperation system was represented as a two-port network, as shown in Figure 2.10, and analyzed using the analogy between mechanical and electrical systems [86, 87]. In the figure, f_h, f_{md}, f_s and f_e are the force applied to the human operator, desired torque for the master manipulator, torque produced by the slave manipulator and force between the slave and environment, respectively. $\dot{q}_h$, $\dot{q}_m$, $\dot{q}_{sd}$ and $\dot{q}_s$ are the desired velocity from the human operator, velocity of the master manipulator, desired velocity of the slave manipulator and actual velocity of the slave manipulator, respectively.

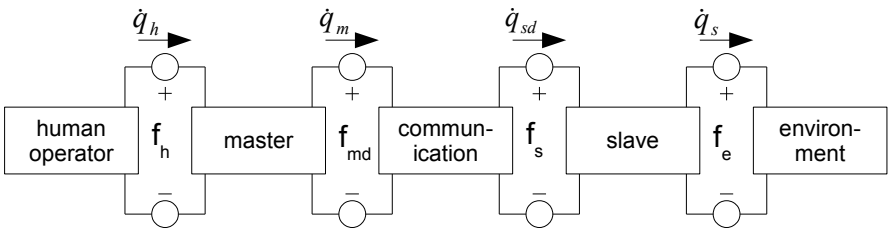


Figure 2.10: Block diagram of a teleoperator system [32].

First, a short description of passivity is presented, before defining the scattering operator. The last section covers the description of scattering variables.

The idea behind the scattering theory is to treat the generalized force and velocities as voltage and current respectively. This makes any passivity analyzis easier, because the effect in an electrical system is simply the product of current and voltage, see Section A.1.

Scattering Operator

For an electrical system, a lossless transmission line, of length l, can be modeled as an infinite series of elements consisting of inductances and capacitances, shown in Figure 2.11.

Figure 2.11: Two-port network [32].

Using a hybrid matrix $H(s)$ to represent a two-port gives [88]

$$\begin{bmatrix} f_1(s) \\ -\dot{q}_2 \end{bmatrix} = H(s) \begin{bmatrix} \dot{q}_1 \\ f_2 \end{bmatrix}, \tag{2.36}$$

where f_i are the generalized forces or voltages, and $\dot{q}_i$ are the velocities or currents. The desired behavior for a teleoperation system is

$$f_{md} = f_s, \qquad \dot{q}_{sd} = \dot{q}_m. \tag{2.37}$$

For an ideal case, the hybrid matrix for the communication network would be

$$H(s) = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}. \tag{2.38}$$

The scattering operator S is defined as

$$f - v = S(f + v), \tag{2.39}$$

and is a mapping from force plus velocity to force minus velocity.

For a two-port network, the scattering matrix relates to the hybrid matrix as

$$\begin{aligned} \begin{bmatrix} f_1(s) - \dot{q}_1(s) \\ f_2(s) + \dot{q}_2(s) \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \left(\begin{bmatrix} f_1(s) \\ -\dot{q}_2(s) \end{bmatrix} - \begin{bmatrix} \dot{q}_1(s) \\ f_2(s) \end{bmatrix} \right) \\ &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} (H(s) - I) \begin{bmatrix} \dot{q}_1(s) \\ f_2(s) \end{bmatrix}. \end{aligned} \tag{2.40}$$

In the same way it can be shown that

$$\begin{bmatrix} f_1(s) + \dot{q}_1(s) \\ f_2(s) - \dot{q}_2(s) \end{bmatrix} = (H(s) + I) \begin{bmatrix} \dot{q}_1(s) \\ f_2(s) \end{bmatrix}. \tag{2.41}$$

Therefore, the scattering matrix is defined as

$$S(s) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} (H(s) - I)(H(s) + I)^{-1}. \tag{2.42}$$

The scattering matrix can be used to prove passivity of a system. A system is passive if and only if the norm of its scattering operator S is less than or equal to one [32]. The proof of this can easily be shown: If $\|S\| \leq 1$ then $\|f - v\|_2 / \|f + v\|_2 \leq 1$, implying that $\|f + v\|_2^2 - \|f - v\|_2^2 \geq 0$. Writing the norm explicitly gives

$$\int_0^\infty (f + v)^T (f + v) - (f - v)^T (f - v)\, dt \geq 0, \tag{2.43}$$

which is equivalent to

$$4 \int_0^\infty f^T v\, dt \geq 0 \quad \Rightarrow \int_0^\infty f^T v\, dt \geq 0, \tag{2.44}$$

showing that the system is passive. Reversing the argument will show necessity.

When time delay is introduced and the same references is used, that is,

$$\begin{aligned} f_{md}(t) &= f_s(t - T) \\ v_{sd}(t) &= v_m(t - T), \end{aligned} \tag{2.45}$$

the hybrid matrix becomes

$$H(s) = \begin{bmatrix} 0 & e^{-sT} \\ -e^{-sT} & 0 \end{bmatrix}, \tag{2.46}$$

giving

$$\begin{aligned} S &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} (H(s) - I)(I + H(s))^{-1} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} -1 & e^{-sT} \\ -e^{-sT} & -1 \end{bmatrix} \begin{bmatrix} 1 & e^{-sT} \\ -e^{-sT} & 1 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} -\tanh(sT) & \operatorname{sech}(sT) \\ \operatorname{sech}(sT) & \tanh(sT) \end{bmatrix}. \end{aligned} \tag{2.47}$$

It can be shown that the norm is given as

$$\|S\| = \sup_{\omega}(|\tan(\omega T)| + |\sec(\omega T)|) = \infty. \tag{2.48}$$

Since the scattering operator is unbounded, the communication network is not passive and stability can therefore not be guaranteed. This shows that the force and velocity references cannot be chosen arbitrarily.

The following input-output relationship in the frequency domain for a lossless transmission line will be used to prove passivity and is given as [89]

$$\begin{aligned} f_1(s) &= Z_0 \tanh(sl/v_0) v_1(s) + \operatorname{sech}(sl/v_0) f_2(s) \\ -v_2(s) &= -\operatorname{sech}(sl/v_0) v_1(s) + (\tanh(sl/v_0)/Z_0) f_2(s), \end{aligned} \tag{2.49}$$

where $Z_0 = \sqrt{L/C}$, $v_0 = 1/\sqrt{LC}$, L is the characteristic inductance, C is the is the capacitance, and l is the length of the transmission line. By setting $Z_0 = 1$ and $v_0 = l/T$, where T is the delay, the scattering operator becomes

$$S(s) = \begin{bmatrix} 0 & e^{-sT} \\ e^{-sT} & 0 \end{bmatrix}, \tag{2.50}$$

where the norm is $\|S\| = 1$, proving that the network is passive. If the human operator is passive as well, this proves that the system is stable.

Scattering Variables

Anderson and Spong (1989b) [90] extended the results, and proposed to use scattering variables. Instead of transmitting the forces and velocities over the communication channel, their corresponding scattering variables are sent [91]. Scattering variables are a combination of the forces, f, and velocities, v, and are defined as

$$\begin{bmatrix} s^+(t,x) \\ s^-(t,x) \end{bmatrix} = \mathbf{T} \begin{bmatrix} f(t,x) \\ v(t,x) \end{bmatrix}, \tag{2.51}$$

where $\mathbf{T}$ is defined as

$$\mathbf{T} = \begin{bmatrix} 1 & Z_0 \\ 1 & -Z_0 \end{bmatrix}. \tag{2.52}$$

$Z_0 = \sqrt{\frac{L}{C}}$ corresponds to the impedance of the transmission line, while $f(t,x)$ and $v(t,x)$ represents the voltage and current in the spatial coordinate $x \in [0,l]$, where l is the length of the virtual line. The scattering variables satisfy

$$\begin{bmatrix} s^+(t,l) \\ s^-(t,l) \end{bmatrix} = \begin{bmatrix} s^+(t-T,0) \\ s^-(t+T,0) \end{bmatrix}, \tag{2.53}$$

where T is the propagation delay and corresponds in an electrical line to $T = l\sqrt{LC}$. L is equivalent to the characteristic inductance, C the capacitance, and l the length of the transmission line. $f(t,0)$ corresponds to $\mathbf{f}_{md}(t)$, $f(t,l)$ to $\mathbf{f}_s(t)$, $v(t,0)$ to $\dot{\mathbf{q}}_m(t)$, and $v(t,l)$ to $\dot{\mathbf{q}}_{sd}(t)$. The object is achieved, transmitting from the master side the signal $s^+(t,0)$ and from the slave side $s^-(t,l)$ and then use (2.51) to reconstruct the voltages and currents. The following relationships are then given

$$s^+(t,l) = \mathbf{f}_s + Z_0\dot{\mathbf{q}}_{sd} \equiv s^+(t-T,0) = \mathbf{f}_{md}(t-T) + Z_0\dot{\mathbf{q}}_m(t-T) \tag{2.54}$$

$$s^-(t,0) = \mathbf{f}_{md} - Z_0\dot{\mathbf{q}}_m \equiv s^-(t-T,l) = \mathbf{f}_s(t-T) - Z_0\dot{\mathbf{q}}_{sd}(t-T). \tag{2.55}$$

Niemeyer, Gunter and Slotine, Jean-Jacques E. (1990) [33] exploited that the transmission line is virtual, selecting the coefficient Z_0 arbitrarily. A

normalized version of (2.51) is given as

$$\begin{bmatrix} \mathbf{s_i^+} \\ \mathbf{s_i^-} \end{bmatrix} = \frac{1}{\sqrt{2b}} \begin{bmatrix} \mathbf{I} & b\mathbf{I} \\ \mathbf{I} & -b\mathbf{I} \end{bmatrix} \begin{bmatrix} \bar{\tau}_i \\ \dot{\mathbf{q}}_{id} \end{bmatrix}, \tag{2.56}$$

where b is the virtual impedance and $\mathbf{I}$ is the $n \times n$ identity matrix.

The aforementioned results are important to ensure passivity in teleoperation systems, and many controllers are based on these results. The next section covers some of the controllers proposed to ensure output synchronization for kinematically identical master and slave manipulator.

2.5.2 Output Synchronization

In this section various controllers proposed by different researches are discussed. Most of the controllers here can also be found in Skumsnes (2011) [45], with some modifications. In addition, the section covering adaptive schemes is extended. The purpose of this discussion is to understand the main principles behind and the advantages and disadvantages of the different controllers. The controllers are organized in three groups: *scattering-based*, *damping injection controllers* and *adaptive schemes*. All groups are based on ensuring passivity (see A.1), which can be used to prove that the controller is stable and that the output energy is bounded with bounded input energy.

The reason to investigate scattering-based controllers is that the basic version was one of the first controllers proven to be passive (and therefore stable) with time delay. Many controllers are based on this result and the basic theory is discussed first. Damping injection controllers are very similar to ordinary independent joint control, found in *Robot Dynamics and Control* [92], which is based on classic proportional-derivative controller (PD-controller), and will be further investigated in following section. The main principle behind adaptive schemes is to identify part of the dynamics with an estimator, presented in the last section. This could be very useful if the dynamics change or is partially unknown. In addition to the controllers mentioned above, there are

controllers that do not deal with time delays, and controllers that have been designed using linearized teleoperation models.

The nonlinear model for the master and slave manipulator, which the controllers are based on, can be written in general form as

$$\begin{aligned} \mathbf{D}_m(\mathbf{q}_m)\ddot{\mathbf{q}}_m + \mathbf{C}_m(\mathbf{q}_m, \dot{\mathbf{q}}_m)\dot{\mathbf{q}}_m + \mathbf{g}_m(\mathbf{q}_m) &= \tau_h - \tau_m^* \\ \mathbf{D}_s(\mathbf{q}_s)\ddot{\mathbf{q}}_s + \mathbf{C}_s(\mathbf{q}_s, \dot{\mathbf{q}}_s)\dot{\mathbf{q}}_s + \mathbf{g}_s(\mathbf{q}_s) &= \tau_s^* - \tau_e, \end{aligned} \tag{2.57}$$

where $\mathbf{q}_i$, $\dot{\mathbf{q}}_i$, $\ddot{\mathbf{q}}_i$ are the joint positions, velocities and accelerations; $\mathbf{D}_i(\mathbf{q}_i)$ are the inertia matrices; $\mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)$ are the Coriolis and centrifugal effects; $\mathbf{g}_i(\mathbf{q}_i)$ are the gravitational forces; τ_i^* are the controllers; and τ_h and τ_e the generalized forces due to the forces exerted by the human and environment. The gravitation is pre-compensated in τ_i^*, giving $\tau_m^* = \tau_m - \mathbf{g}_m(\mathbf{q}_m)$ and $\tau_s^* = \tau_s + \mathbf{g}_s(\mathbf{q}_s)$.

To make the comparison between the different controllers easier, the same variables are used for each controller. Table 2.1 lists the variables and parameters, with a short description of their interpretation, used in the derivations.

Table 2.1: List of variables and parameters

	Variables		Parameters	
human operator	τ_h	applied force		
master	$\bar{\tau}_m$	coordinating torque	$\mathbf{D}_m$	inertia
	τ_m	motor torque	B_m	rate damping
	$\mathbf{q}_m$	position	K_{dm}	error damping
	$\dot{\mathbf{q}}_m$	velocity	K_m	stiffness gain
commun. block	$\dot{\mathbf{q}}_{sd}$	desired slave vel.	T	time delay
	$\dot{\mathbf{q}}_{md}$	delayed slave vel.	n	scale factor
slave	$\bar{\tau}_s$	coordinating torque	$\mathbf{D}_s$	inertia
	τ_s	motor torque	B_s	rate damping
	$\mathbf{q}_s$	position	K_{ds}	error damping
	$\dot{\mathbf{q}}_s$	velocity	K_s	stiffness gain
environment	τ_e	contact force	Z_e	impedance
			α_f	force gain

Scattering-based schemes

Scattering-based schemes are based on the principles described in Section 2.5.1, ensuring passivity by controlling the virtual impedance.

In addition to a control law for the communication circuit, Anderson and Spong (1989a) [32] propose to complete the design with a PI controller on the remote side, and damping injection terms on both sides. A damping injection term is an effect that reduces the velocity. By using Lyapunov stability analysis, it is proven that the controller ensures an asymptotically stable behavior of the local and remote velocities, despite constant time-delays [90]. The control strategy for the master and slave is

$$\tau_m = B_m \dot{\mathbf{q}}_m + \bar{\tau}_m \tag{2.58}$$

$$\tau_s = \bar{\tau}_s - B_s \dot{\mathbf{q}}_s - \alpha_f \tau_e, \tag{2.59}$$

where $\dot{\mathbf{q}}_m$ and $\dot{\mathbf{q}}_s$ are the master and slave velocities respectively, and τ_m and τ_s are the respective motor torques. Furthermore, B_m and B_s are the master and slave rate damping, α_f is the environment force gain, $\bar{\tau}_m$ is the desired force for the master manipulator, and τ_e is the environment torque. $\bar{\tau}_s$ is the coordinating torque for the slave manipulator, given by

$$\bar{\tau}_s = -K_s \int (\dot{\mathbf{q}}_s - \dot{\mathbf{q}}_{sd}) dt - K_{ds}(\dot{\mathbf{q}}_s - \dot{\mathbf{q}}_{sd}), \tag{2.60}$$

where $\dot{\mathbf{q}}_{sd}$ and K_{ds} are the desired velocity and error damping for the slave respectively. The desired values for the slave velocities, $\dot{\mathbf{q}}_{sd}$, and master force, $\bar{\tau}_m$, are found by solving the equations for the scattering variables, (2.54) and (2.55), giving the following equations

$$\bar{\tau}_m(t) = \bar{\tau}_s(t-T) + Z_0[\dot{\mathbf{q}}_m(t) - \dot{\mathbf{q}}_{sd}(t-T)] \tag{2.61}$$

$$\dot{\mathbf{q}}_{sd}(t) = \dot{\mathbf{q}}_m(t-T) + \frac{1}{Z_0}[\bar{\tau}_m(t-T) - \bar{\tau}_s(t)]. \tag{2.62}$$

If the force and velocity signal differ by orders of magnitude, the control law given by (2.61) and (2.62) may have implementation problems. This

potential problem was solved by multiplying the impedance with a scaling factor, n^2, in the control law, that is,

$$\bar{\tau}_m(t) = \bar{\tau}_s(t-T) + n^2 Z_0[\dot{\mathbf{q}}_m(t) - \dot{\mathbf{q}}_{sd}(t-T)] \tag{2.63}$$

$$\dot{\mathbf{q}}_{sd}(t) = \dot{\mathbf{q}}_m(t-T) + \frac{1}{n^2 Z_0}[\bar{\tau}_m(t-T) - \bar{\tau}_s(t)]. \tag{2.64}$$

This type of scheme is called *Classical scattering scheme* and can handle constant time-delay but not variable time-delay or position tracking. This is the first of several scattering-based schemes.

When the impedance of a physical transmission line is different from the impedance of the line termination, wave reflection can occur. Wave reflection is a phenomenon in the transmission lines that deforms the transmitted signals, and degrades the performance. The same affect can be shown to occur for scattering variables. Substituting the equation for the coordinating torque (2.60) in the expression for the received scattering variable at the slave manipulator (2.54), gives

$$s^+(t,l) = -K_s \int (\dot{\mathbf{q}}_s - \dot{\mathbf{q}}_{sd})dt - K_{ds}(\dot{\mathbf{q}}_s - \dot{\mathbf{q}}_{sd}) + Z_0 \dot{\mathbf{q}}_{sd}. \tag{2.65}$$

By inserting (2.65) into the definition of scattering variables, (2.51) and (2.53), yields the delay differential equation

$$\dot{\mathbf{q}}_{sd} + \frac{Z_0 - K_{ds}}{Z_0 + K_{ds}} \dot{\mathbf{q}}_{sd}(t-2T) = g(\textstyle\int \dot{\mathbf{q}}_{sd}, \dot{\mathbf{q}}_m), \tag{2.66}$$

where g is some functional relation. If the reflection coefficient $\frac{Z_0 - K_{ds}}{Z_0 + K_{ds}}$ is different from zero, $\dot{\mathbf{q}}_{sd}$ exhibits large oscillations, corresponding to physical wave reflections. In *Symmetric impedance matching*, proposed by Anderson and Spong (1989b) [90] and Niemeyer, Gunter and Slotine, Jean-Jacques E. (1990) [33], the main idea is to select $K_{ds} = Z_0$ to make this effect disappear, add a PI action on the master manipulator, and use the normalized implementation of (2.51), (2.56), where b is the virtual impedance. The new

controller is

$$\begin{aligned} \bar{\tau}_m &= K_m \int (\dot{\mathbf{q}}_m - \dot{\mathbf{q}}_{md}) dt + K_{dm}(\dot{\mathbf{q}}_m - \dot{\mathbf{q}}_{md}) \\ \bar{\tau}_s &= -K_s \int (\dot{\mathbf{q}}_s - \dot{\mathbf{q}}_{sd}) dt + K_{ds}(\dot{\mathbf{q}}_s - \dot{\mathbf{q}}_{sd}). \end{aligned} \tag{2.67}$$

Here K_{dm} and K_m correspond to the same variables on the master side as K_{ds} and K_s on the slave side. Symmetric impedance matching scheme can handle constant time-delay, but can't handle position tracking or variable time-delay.

The principle of *Position tracking controllers* is to send explicit position information of each manipulator together with the scattering variables. This controller was proposed by Chopra, Spong, Ortega, and Nikita E. (2006) [21],

$$\tau_m = \bar{\tau}_m + K\mathbf{e}_m + B_m\dot{\mathbf{q}}_m, \quad \tau_s = \bar{\tau}_s - K\mathbf{e}_s - B_s\dot{\mathbf{q}}_s, \tag{2.68}$$

where

$$\mathbf{e}_m = \mathbf{q}_m - \mathbf{q}_s(t-T) \quad \text{and} \quad \mathbf{e}_s = \mathbf{q}_s - \mathbf{q}_m(t-T), \tag{2.69}$$

and

$$\bar{\tau}_s = -K_{ds}(\dot{\mathbf{q}}_s - \dot{\mathbf{q}}_{sd}). \tag{2.70}$$

The controller gain is chosen such that $K = \frac{K_m}{b} = \frac{K_s}{b}$, where b is the virtual impedance. The position is tracked, and the controller can handle constant, but not variable, time-delay.

Namerikawa and Kawada (2006)[93] proposed *Symmetric position tracking*, a scheme aimed at eliminating the wave reflections with the use of a symmetric controller and by matching the impedances. The new controller is given as in (2.68), but with

$$\bar{\tau}_m = K_{dm}(\dot{\mathbf{q}}_m - \dot{\mathbf{q}}_{md}) \quad \bar{\tau}_s = -K_{ds}(\dot{\mathbf{q}}_s - \dot{\mathbf{q}}_{sd}), \tag{2.71}$$

and the control gain such that

$$2B_mB_s > (T_m^2 + T_s^2)K^2. \tag{2.72}$$

In this controller, the scattered velocity expressions do not contain any double delayed term. Symmetric position tracking tracks the position and works with constant time-delay, but not variable time-delay.

To deal with variable time-delay, Lozano, Chopra and Spong (2002) [94] suggest to use a time-varying gain γ_i in the connection between the master and slave manipulator. The reason for this variable is that the scattering transformation (2.56) is not passive when the time-delay is variable. The new scattering variables that are suggested are

$$\mathbf{s_s^+} = \gamma_\mathbf{m}\mathbf{s_m^+}(\mathbf{t} - \mathbf{T_m}(\mathbf{t})); \quad \mathbf{s_m^-} = \gamma_\mathbf{s}\mathbf{s_s^-}(\mathbf{t} - \mathbf{T_s}(\mathbf{t})). \tag{2.73}$$

This scheme is called *Classic scattering for variable time-delays* and are stable with both constant and variable time-delay, but can not guarantee position tracking.

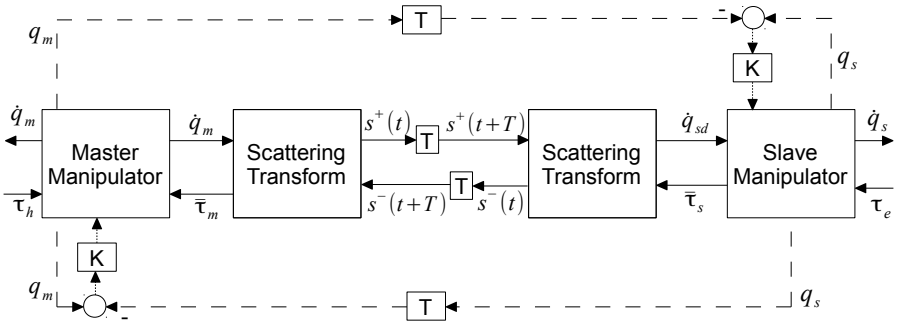


Figure 2.12: General scheme for scattering-based controllers [21]. Classic scattering scheme, symmetric impedance matching and classic scattering for variable time-delays are represented with the dotted lines disabled, while position tracking controller, symmetric position tracking and position tracking for variable time-delays are represented with dotted lines enabled.

To achieve position tracking for variable time-delay Nuño et al.(2009) [95] proposed a new controller

$$\begin{aligned} \tau_m &= \bar{\tau}_m + K\epsilon_m + B_m\dot{\mathbf{q}}_m \\ \tau_s &= \bar{\tau}_s - K\epsilon_s - B_s\dot{\mathbf{q}}_s, \end{aligned} \tag{2.74}$$

where

$$\epsilon_m = \mathbf{q}_m - \mathbf{q}_s(t - T_s(t)) \quad \text{and} \quad \epsilon_s = \mathbf{q}_s - \mathbf{q}_m(t - T_m(t)), \tag{2.75}$$

and $\bar{\tau}_i$ is given by (2.71). The scattered velocities are codified using (2.55), (2.54), and (2.73) with $\gamma_i^2 = 1 - \dot{T}_i(t)$. This scheme, called *Position tracking for variable time-delays*, provides stability with constant and variable time-delay and can guarantee position tracking. [96] shows that the performance of this controller is improved by using the impedance matching of [33], choosing $K_{dm} = K_{ds} = b$ in (2.71).

Figure 2.12 shows a general representation of scattering-based schemes. The version with the dotted lines disabled can be used for classic scattering scheme, symmetric impedance matching and classic scattering for variable time-delays, while the version with the dotted lines enabled can be used for position tracking controller, symmetric position tracking and position tracking for variable time-delays. The schemes represented with the dotted lines enable are the only ones guaranteeing position tracking.

Damping injection schemes

Another type of schemes are *Damping injection schemes*. These are passivity based controllers for manipulators that make it possible to obtain asymptotic stability [97]. Lee and Spong (2006) [98] propose to use a *Proportional-Derivative + damping controller* (PD + d controller) for teleoperators with constant time-delays, which was proven to be stable, first by Nuño, Ortega, Barabanov and Basanez (2008) [99]. Nuño, Basañez,Ortega and Spong (2008) [100] and Nuño, Basañez, Ortega and Spong (2009) [95] showed that the PD+d controllers are stable with variable time-delays,

$$\begin{aligned} \tau_m &= K_d[\dot{\mathbf{q}}_m - \gamma_s \dot{\mathbf{q}}_s(t - T_s(t))] + K_m \epsilon_m + B_m \dot{\mathbf{q}}_m \\ \tau_s &= -K_d[\dot{\mathbf{q}}_s - \gamma_m \dot{\mathbf{q}}_m(t - T_m(t))] - K_s \epsilon_s - B_s \dot{\mathbf{q}}_s, \end{aligned} \tag{2.76}$$

where K_d,K_i,B_m,B_s $\in \mathbb{R}_{>0}$, γ_i are defined by $\gamma_i^2(t) = 1 - \dot{T}_i(t)$ and ϵ_i is defined in (2.75).

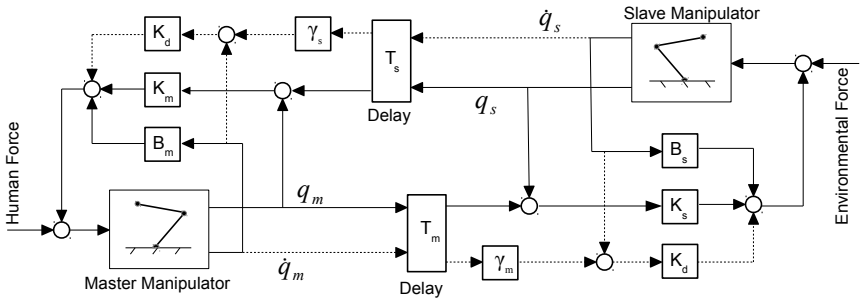


Figure 2.13: General scheme for damping injection controllers [96]. P + d controller is represented with the dotted lines disabled, while the PD + d controller is represented with dotted lines enabled.

Because PD + d controllers use γ_i, which depends on the rate of change of the delays, the controllers are sensitive to abrupt changes in time-delay. The simpler *Proportional + damping controller* (P + d controller) does not make use of this variable gain, which may improve the performance. The new controller is given by,

$$\tau_m = K_m\epsilon_m + B_m\dot{\mathbf{q}}_m; \quad \tau_s = -K_s\epsilon_s - B_s\dot{\mathbf{q}}_s, \tag{2.77}$$

where the variables are given as in PD + d controller. Both PD + d and P + d ensure position tracking, and are stable with both constant and variable time-delay. A P + d and PD + d controller are shown in Figure 2.13 with the dotted lines disabled and enabled, respectively.

The *Passive output interconnection* schemes are a special case of the PD + d controller, with only the D-action. The idea is to interconnect the delayed passive outputs v_i of the master and slave manipulator [101, 102, 103, 95, 104]. These schemes have delay-independent stability properties. The controller

can be written as,

$$\begin{aligned}\tau_m &= K_d[\dot{\mathbf{q}}_m - \gamma_s \dot{\mathbf{q}}_s(t - T_s(t))] \\ \tau_s &= -K_d[\dot{\mathbf{q}}_s - \gamma_m \dot{\mathbf{q}}_m(t - T_m(t))],\end{aligned} \tag{2.78}$$

where $\gamma_i^2(t) = 1 - \dot{T}_i(t)$ and $K_d \in \mathbb{R}_{>0}$. These schemes are stable under constant and variable time-delay, but do not originally guarantee position tracking. Chopra and Spong (2006) [101] proposed a model-based controller, making these schemes provide position tracking,

$$\begin{aligned}\tau_m &= K[\mathbf{r}_m - \mathbf{r}_s(t - T_s)] + \mathbf{D}_m(\mathbf{q}_m)\lambda\dot{\mathbf{q}}_m + \mathbf{C}_m(\mathbf{q}_m, \dot{\mathbf{q}}_m)\lambda\mathbf{q}_m \\ \tau_s &= K[\mathbf{r}_m(t - T_m) - \mathbf{r}_s] - \mathbf{D}_s(\mathbf{q}_s)\lambda\dot{\mathbf{q}}_s - \mathbf{C}_s(\mathbf{q}_s, \dot{\mathbf{q}}_s)\lambda\mathbf{q}_s,\end{aligned} \tag{2.79}$$

where $\mathbf{r}_i = \dot{\mathbf{q}}_i + \lambda\mathbf{q}_i$; $\lambda = \lambda^T > 0$; $\mathbf{D}_i(\mathbf{q}_i) \in \mathbb{R}^{n\times n}$ are the inertia matrices; and $\mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i) \in \mathbb{R}^{n\times n}$ are the Coriolis and centrifugal effects from the model in (2.57).

Adaptive schemes

Adaptive schemes for nonlinear telerobotics are similar to standard adaptive manipulator controllers [105], and are based on two assumptions: that the model is linearly dependent on the parameters θ_i, and that the operator defined by the parameter update law $\dot{\hat{\theta}}_i$ are passive, where $\hat{\theta}_i$ is the estimation of θ_i. The update law for the estimators is given as

$$\dot{\hat{\theta}}_i = \mathbf{\Gamma}_i \mathbf{Y}_i^T \nu_i, \tag{2.80}$$

where $\mathbf{\Gamma}_i$ are constant, symmetric and positive definite matrices.

Chopra, Spong and Lozano (2008) [38] propose a control scheme with ν_i defined as

$$\nu_i = \dot{\mathbf{q}}_i + \lambda\mathbf{q}_i, \tag{2.81}$$

where $\lambda > 0$ is a diagonal matrix. The new controllers are given as

$$\begin{aligned} \tau_m^* &= \mathbf{Y}_m(\mathbf{q}_m, \dot{\mathbf{q}}_m)\hat{\theta}_m + \bar{\tau}_m \\ \tau_s^* &= -\mathbf{Y}_s(\mathbf{q}_s, \dot{\mathbf{q}}_s)\hat{\theta}_s - \bar{\tau}_s, \end{aligned} \tag{2.82}$$

where $\mathbf{Y}_i\hat{\theta}_i = \hat{\mathbf{D}}_i(\mathbf{q}_i)\lambda\ddot{\mathbf{q}}_i + \hat{\mathbf{C}}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\lambda\dot{\mathbf{q}}_i - \hat{\mathbf{g}}_i(\mathbf{q}_i)$. Substituting the equation for the new controller (2.82) into the model (2.57) leads to

$$\begin{aligned} \mathbf{D}_m(\mathbf{q}_m)\dot{\nu}_m + \mathbf{C}_m(\mathbf{q}_m, \dot{\mathbf{q}}_m)\nu_m &= \mathbf{Y}_m\tilde{\theta}_m - \bar{\tau}_m + \tau_h \\ \mathbf{D}_s(\mathbf{q}_s)\dot{\nu}_s + \mathbf{C}_s(\mathbf{q}_s, \dot{\mathbf{q}}_s)\nu_s &= \mathbf{Y}_s\tilde{\theta}_s - \bar{\tau}_s - \tau_e, \end{aligned} \tag{2.83}$$

with $\tilde{\theta}_i = \theta_i - \hat{\theta}_i$. The coordinating torques, $\bar{\tau}_i$, are suggested to be

$$\begin{aligned} \bar{\tau}_m &= \mathbf{K}_m(\nu_m - \nu_s(t-T)) \\ \bar{\tau}_s &= \mathbf{K}_s(\nu_s - \nu_m(t-T)), \end{aligned} \tag{2.84}$$

where $\mathbf{K}_i = \mathbf{K}_i^T > 0$ are constant matrices.

Miller, Lee and Krovi (2009) [106] propose to use a design based on Chopra, Spong and Lozano (2008) [38]. A control design with $\nu_i = \dot{\mathbf{e}}_i + \lambda\mathbf{e}_i$ is first presented, where the errors $\mathbf{e}_i$ are defined as previously:

$$\mathbf{e}_m = \mathbf{q}_m - \mathbf{q}_s(t-T) \quad \text{and} \quad \mathbf{e}_s = \mathbf{q}_s - \mathbf{q}_m(t-T). \tag{2.85}$$

When the desired trajectory is the origin, such that $\mathbf{e}_i = \mathbf{q}_i$, the design is equal to Chopra, Spong and Lozano (2008) [38].

A design similar to the scheme of Miller, Lee and Krovi (2009) [106], also based on the scheme of Chopra, Spong and Lozano (2008) [38], is presented by Nuño, Ortega and Basañez (2010) [39]. Here ν_i is defined as

$$\nu_i = \dot{\mathbf{q}}_i + \lambda\mathbf{e}_i, \tag{2.86}$$

with $\mathbf{e}_i$ defined in (2.69), and where $\lambda > 0$ is a diagonal matrix. The con-

trollers are equal to (2.82), but with

$$\mathbf{Y}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i, \mathbf{e}_i, \dot{\mathbf{e}}_i)\hat{\theta}_i = \hat{\mathbf{D}}_i(\mathbf{q}_i)\lambda\dot{\mathbf{e}}_i + \hat{\mathbf{C}}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\lambda\mathbf{e}_i - \hat{\mathbf{g}}_i(\mathbf{q}_i), \tag{2.87}$$

and the coordinating torques suggested to be $\bar{\tau}_i = \mathbf{K}_i\nu_i + \mathbf{B}\dot{\mathbf{e}}_i$, where $\mathbf{K}_i = \mathbf{K}_i^T > 0$ and $\mathbf{B}$ is diagonal and positive definite.

Chopra and Spong (2004) [107] propose an adaptive controller using scattering variables. By encoding the variables ν_i instead of the velocities $\dot{q}$, the new scattering variables are given as

$$\begin{bmatrix} \mathbf{s_i^+} \\ \mathbf{s_i^-} \end{bmatrix} = \frac{1}{\sqrt{2b}} \begin{bmatrix} \mathbf{I} & b\mathbf{I} \\ \mathbf{I} & -b\mathbf{I} \end{bmatrix} \begin{bmatrix} \bar{\tau}_i \\ \nu_{id} \end{bmatrix}, \tag{2.88}$$

where $\bar{\tau}_i = K(\nu_{id} - \nu_i)$. This version of adaptive schemes, called *Scattering-based state synchronization*, can guarantee stability with variable time-delays.

The different aforementioned controllers are limited to identical master-slave system. One problem that arises is how to translate the movement of a human operated joystick, with limited range of motion, to a mobile platform with, in theory, infinite range of motion. The next section will therefore discuss different solutions proposed when handling mobile robots.

2.5.3 Force Rendering Scheme

A special case of teleoperation is the use of mobile robots, and handling of obstacles. Experiments done in by Diolaiti and Melchiorri (2002)[41] and Lee Sukhatme, Kim, and Park (2002) [108] show that the task error in controlling mobile robots is reduced by introducing haptic feedback. One area of research is the force rendering, the process of computing the force that the operator feels. A short description of the different schemes will follow in the next subsections, and are based on the work done in Skumsnes (2011) [45].

Potential Force

In Khatib (1986) [109] a potential force field is used for path planning of mobile robots. This is a obstacle avoidance approach with the use of analytic primitives for geometric modeling, but the path planning and geometric modeling require knowledge about every obstacle between the mobile robot and the goal.

Spring Force

Lee Sukhatme, Kim, and Park (2002) [108] introduce an environmental and a collision-preventing force, which differs from the force field in Khatib (1986) [109] since there is no attraction to a goal, and only obstacles in the area where the robot may reach in near future are considered. The distance is measured by a laser scanner mounted on the robot, while in Roth, Schilling, and Rösch (2002) [43] the contact force is measured with a force sensor. The measurements are stored in a vector R, and the difference between R and a fixed vector R_o is called Δ. That is,

$$R = (r_1 \quad r_2 \quad ... \quad r_n) \tag{2.89}$$

$$R_o = (r_{o1} \quad r_{o2} \quad ... \quad r_{on}) \tag{2.90}$$

$$\Delta = R_o - R = (\delta_1 \quad \delta_2 \quad ... \quad \delta_n), \tag{2.91}$$

where r_i is the measured distance from the i^{th} sensor and r_{oi} is the fixed distance from where generation of feedback starts. The force is inversely proportional to the distance to the obstacle, and is calculated as

$$f_i = \begin{cases} k_i\delta_i, & r_i \leq r_{oi} \\ 0, & r_i > r_{oi} \end{cases} \tag{2.92}$$

$$F = (f_1 \quad f_2 \quad ... \quad f_n) \tag{2.93}$$

where f_i is the relevant force feedback equal to the distance δ_i, measured from the i^{th} sensor multiplied with the virtual stiffness k_i.

Spring-Damper Force

As in Lee Sukhatme, Kim, and Park (2002) [108], Diolaiti and Melchiorri (2002) [41] use sensors to build a local map of the surrounding obstacles, but Diolaiti and Melchiorri (2002) [41] emulates a physical contact by a virtual spring and damper.

Variable gain

In Farkhatdinov, Ryu, and An (2010) [110] another force rendering approach, with variable feedback gain, is proposed. Here they modify the stiffness k in (2.92), to be based on R and dR/dt. The variable gain k_i^* is based on the distance measured from the i^{th} sensor,

$$k_i^* = \begin{cases} k_{min}, & \frac{dr_i}{dt} \geq 0 \\ \frac{1}{\gamma}(k_{max} - k_{min})\frac{dr_i}{dt} + k_{min}, & -\gamma < \frac{dr_i}{dt} < 0 \\ k_{max}, & \frac{dr_i}{dt} \leq -\gamma \end{cases} \tag{2.94}$$

where k_{min} and k_{max} are the minimum and maximum marginal values of the feedback gain and γ is the boundary of the speed between the mobile robot and the obstacle.

2.5.4 Controllers for Mobile Manipulators

The theories, designs and results presented in the preceding sections, in 2.5.2 and in 2.5.3, are used by various researchers to design different control architectures for mobile manipulators.

As mentioned earlier, the nonlinear model for a manipulator can be written as

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \tau, \tag{2.95}$$

where $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ are the joint positions, velocities and accelerations; $\mathbf{D}(\mathbf{q})$ are the inertia matrices; $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ are the Coriolis and centrifugal effects; $\mathbf{g}(\mathbf{q})$ are

the gravitational forces; and τ the generalized forces acting on the system. For a system with nonholonomic constraints the equations of motion are given by

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{B}(\mathbf{q})\tau + \mathbf{G}(\mathbf{q})^T\lambda \tag{2.96}$$

where $\mathbf{B}(\mathbf{q})$ is the map from the actuation space to the extended coordinate space, λ are the constraint forces, and $\mathbf{G}(\mathbf{q})$ is the constraint matrix, satisfying

$$\mathbf{G}(\mathbf{q})\dot{\mathbf{q}} = 0 \qquad \text{and} \qquad \mathbf{G}(\mathbf{q})\mathbf{M}(\mathbf{q}) = 0,$$

where $\mathbf{M}(\mathbf{q})$ is a mapping from pseudo-velocities, $\mathbf{u}$, to generalized velocities, $\dot{\mathbf{q}}$. By exploiting the following relationships:

$$\dot{\mathbf{q}} = \mathbf{M}(\mathbf{q})\mathbf{u} \qquad \text{and} \qquad \ddot{\mathbf{q}} = \mathbf{M}(\mathbf{q})\dot{\mathbf{u}} + \dot{\mathbf{M}}(\mathbf{q})\mathbf{u},$$

the constrained system can be projected into the feasible motion subspace, written as

$$\mathbf{H}(\mathbf{q})\dot{\mathbf{u}} + \mathbf{V}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{u} + \mathbf{R}(\mathbf{q}) = \mathbf{N}(\mathbf{q})\tau, \tag{2.97}$$

with the following connection

$$\begin{aligned} \mathbf{H}(\mathbf{q}) &= \mathbf{M}(\mathbf{q})^T\mathbf{D}(\mathbf{q})\mathbf{M}(\mathbf{q}) \\ \mathbf{V}(\mathbf{q}, \dot{\mathbf{q}}) &= \mathbf{M}(\mathbf{q})^T\mathbf{D}(\mathbf{q})\dot{\mathbf{M}}(\mathbf{q}) + \mathbf{M}(\mathbf{q})^T\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{M}(\mathbf{q}) \\ \mathbf{R}(\mathbf{q}) &= \mathbf{M}(\mathbf{q})^T\mathbf{g}(\mathbf{q}) \\ \mathbf{N}(\mathbf{q}) &= \mathbf{M}(\mathbf{q})^T\mathbf{B}(\mathbf{q}). \end{aligned} \tag{2.98}$$

Here $\mathbf{H}$, $\mathbf{V}$, and $\mathbf{R}$ capture the inertia, Coriolis and centrifugal effect, and gravity, respectively, while $\mathbf{N}$ maps the generalized torque to the corresponding pseudo-acceleration.

There are, in general, two types of approaches for the control of mobile manipulators [111]: decentralized control and centralized control. When the decentralized control approach is used, the mobile platform and manipulator arm are controlled separately, neglecting the dynamic interaction between them. In the centralized control approach, the mobile manipulator is re-

garded as a redundant robot, where the redundancy is introduced by the motion of mobile platform.

To control the heading of the vehicle, the transformation from the joystick is merely a position-position coordination problem, where many passivity-enforcing schemes are applicable [98, 32, 112]. For the velocity, this is not the case since it requires position-velocity coordination. The reason is that those passivity-enforcing shemes are not capable of coordinating signals with different relative degrees, when force are considered as input. By modifying the energetic passivity [113], one can consider the position of the joystick as velocity [68], and use the listed passivity-enforcing schemes.

Similarly to the theory presented in Section 2.1, Bayle, Fourquet, and Renaud (2001) [54] suggest to exploit the null-space. The main target of the controller is to ensure tracking of the velocity, while the secondary target is to increase the manipulability, giving the following coordination strategy

$$\mathbf{u} = \bar{\mathbf{J}}^{\dagger}(\mathbf{q})\dot{\mathbf{x}} - W(\mathbf{I} - \bar{\mathbf{J}}^{\dagger}(\mathbf{q})\bar{\mathbf{J}}(\mathbf{q}))\left(\frac{\partial \mathcal{P}(\mathbf{q})}{\partial \mathbf{q}}\mathbf{M}\right)^{T}, \tag{2.99}$$

where W is a positive weighting matrix, $\mathcal{P}(\mathbf{q})$ a scalar function, and $\bar{\mathbf{J}}^{\dagger}(\mathbf{q})$ the pseudoinverse of $\bar{\mathbf{J}}(\mathbf{q})$. For a mobile manipulator the function, $\mathcal{P}(\mathbf{q})$, can be written in general form as

$$\mathcal{P}(\mathbf{q}) = \alpha(\mathbf{q})\mathcal{P}_{b+a}(\mathbf{q}) + (1 - \alpha(\mathbf{q}))\mathcal{P}_{a}(\mathbf{q}_a), \tag{2.100}$$

where $\mathcal{P}_a(\mathbf{q}_a)$ and $\mathcal{P}_{b+a}(\mathbf{q})$ are functions based on the manipulability of the arm itself and mobile manipulator in total, respectively. The smooth scalar function $\alpha(\mathbf{q}) \in [0\ 1]$ makes it possible to adapt the choice of criteria based on the configuration of the mobile manipulator. One way of designing $\mathcal{P}(\mathbf{q})$ is to use the manipulability measure presented in Section 2.2, for instance w^{-1} or w_5.

Park and Khatib (2004) [114] propose a controller based on the theory in Section 2.1. Using the operational space formulation, the equations of motion

for the end-effector of the slave manipulator can be written as

$$\mathbf{\Lambda}_s(\mathbf{q}_s)\ddot{\mathbf{x}}_s + \mathbf{\Gamma}_s(\mathbf{q}_s, \dot{\mathbf{q}}_s) + \eta_s(\mathbf{q}_s) = \mathbf{F}_s - \mathbf{F}_e, \tag{2.101}$$

where $\mathbf{x}_s$, $\mathbf{\Lambda}_s(\mathbf{q}_s)$, $\mathbf{\Gamma}_s(\mathbf{q}_s, \dot{\mathbf{q}}_s)$ and $\eta_s(\mathbf{q}_s)$ are the position, inertia matrix, vector of Coriolis and centripetal forces, and the gravity vector for the slave manipulator in operational space, respectively. The forces $\mathbf{F}_s$ and $\mathbf{F}_e$ are applied by the controller and manipulated environment. An estimation of the matrices, denoted $\hat{\cdot}$, is used to eliminate the internal dynamic, giving the control torque as

$$\begin{aligned} \tau_s^* &= \mathbf{J}_s(\mathbf{q}_s)\mathbf{F}_s + \mathbf{N}_s(\mathbf{q}_s)^T\tau_{s0} \\ \mathbf{F}_s &= \hat{\mathbf{\Lambda}}_s\mathbf{F}_s^* + \hat{\mathbf{\Gamma}}_s + \hat{\eta}_s + \hat{\mathbf{F}}_e, \end{aligned} \tag{2.102}$$

where $\mathbf{J}_s(\mathbf{q}_s)$ satisfy $\dot{\mathbf{x}}_s = \mathbf{J}_s(\mathbf{q}_s)\dot{\mathbf{q}}_s$, $\mathbf{N}_s(\mathbf{q}_s)$ is the null-space matrix, and $\mathbf{F}_s^*$ is the command to the unit point mass system. The command $\mathbf{F}_s^*$ consists of force and motion control components, projected by the selection matrices, Ω_f and Ω_m, giving

$$\mathbf{F}_s^* = \mathbf{\Omega}_f\mathbf{F}_{fs}^* + \mathbf{\Omega}_m\mathbf{F}_{ms}^*. \tag{2.103}$$

The master manipulator is modeled as mass-damper system

$$\mathbf{D}_m\ddot{\mathbf{x}}_m + \mathbf{C}_m\dot{\mathbf{x}}_m = \mathbf{F}_h - \mathbf{F}_m, \tag{2.104}$$

where $\mathbf{D}_m$, $\mathbf{C}_m$ and $\mathbf{x}_m$ are the mass, damping effect, and position of the joystick, respectively. $\mathbf{F}_h$ is the force applied by the human operator, while $\mathbf{F}_m$ is the controller and given as

$$\mathbf{F}_m = s_f\mathbf{F}_d, \tag{2.105}$$

where s_f is a force scaling. $\mathbf{F}_d$ the desired contact force for the slave robot, given by

$$\mathbf{F}_d = K_{vir}(s_p\mathbf{x}_m - \mathbf{x}_s(t - T)), \tag{2.106}$$

where s_p is a position scaling, while K_{vir} is a virtual spring between the

master and slave end-effector, used to generate the desired force, $\mathbf{F}_d$. The contact force $\mathbf{F}_e$ is modeled as a spring with a certain stiffness, K_s,

$$\dot{\mathbf{F}}_e = K_s \dot{\mathbf{x}}_s, \tag{2.107}$$

which leads to the equations of motion of contact force in operational space:

$$\ddot{\mathbf{F}}_e = K_s \mathbf{F}_s^*. \tag{2.108}$$

For telepresence, it is important to maintain $K_{vir} \gg K_s$, while the value of K_{vir} is limited by the stability. It is suggested to use adaptive estimation of the stiffness, K_s, to improve the force control and to modify the virtual spring, K_{vir}, to provide a better telepresence to the human operator.

Chen, Liu, Zhang and Rong (2006) [111] propose a control approach which combines elements of the decentralized and centralized controller. This approach uses sub-models, describing the mobile platform and the manipulator, that are derived from the unified dynamic model of the mobile manipulator. Based on this, the mobile manipulator sub-controllers are divided into two parts, controlling the mobile platform and manipulator separately.

Farkhatdinov, Ryu and Poduraev (2008)[61] suggest a control strategy for a manipulator with one DOF, mounted on a mobile platform. The main idea is to split the movement intp two modes: when the human-operator controls the platform's speed and when the human-operator controls the manipulator's position. The generalized forces for the slave manipulator, τ_s, can be separated in to two parts: one part controlling the moving platform, denoted τ_{sp}, and one part controlling the joints for the manipulator, called τ_{sm}. The same partition can be done for the generalized coordinates

$$\mathbf{q}_s = \begin{bmatrix} \mathbf{q}_{sp} \\ \mathbf{q}_{sm} \end{bmatrix}. \tag{2.109}$$

For position control of the slave manipulator, a P+d scheme is used. The control law for speed control of the mobile platform is a passive output in-

terconnection scheme giving the total controller

$$\begin{aligned}\tau_{sm} &= K_s^m(\mathbf{q}_{sm}^{des} - \mathbf{q}_{sm}) - B_s^m\dot{\mathbf{q}}_{sm} \\ \tau_{sp} &= (1 - S_{mode})K_d[\dot{\mathbf{q}}_{sp}^{des} - \dot{\mathbf{q}}_{sp}] + S_{mode}[K_s^p(\mathbf{q}_{sp}^{des} - \mathbf{q}_{sp}) + B_s^p\dot{\mathbf{q}}_{sp}],\end{aligned} \tag{2.110}$$

where $\mathbf{q}_{sm}^{des}$ is the desired position of the manipulator; $\dot{\mathbf{q}}_{sp}^{des}$ and $\mathbf{q}_{sp}^{des}$ are the desired speed and position of the moving platform; and K_s^m, B_s^m, K_d, K_s^p and B_s^p are controller gains. The variable S_{mode} is given as

$$S_{mode} = \begin{cases} 0, & \text{Speed control of platform} \\ 1, & \text{Position control of manipulator.} \end{cases} \tag{2.111}$$

The values for the desired position are based on the master position, and are given as

$$\mathbf{q}_{sm}^{des} = \eta\mathbf{q}_m \quad \dot{\mathbf{q}}_{sp}^{des} = \beta\mathbf{q}_m, \tag{2.112}$$

where η and β are scaling factors. The force feedback to the master manipulator, τ_m, is defined as

$$\tau_m = (1 - S_{mode})\lambda\tau_e + S_{mode}\mu\tau_{sp}, \tag{2.113}$$

where τ_e is the environmental force, modeled as a spring, and λ and μ are scaling coefficients.

In addition to an adaptive scheme for teleoperation system, Miller, Lee and Krovi (2009) [106] present a control scheme for a wheeled mobile robot. If the haptic device is modeled as a translational mass, given as

$$\mathbf{\Lambda}_m(\mathbf{q}_m)\ddot{\mathbf{x}}_m = \mathbf{F}_h - \mathbf{F}_m, \tag{2.114}$$

and the mobile robot is modeled using the feasible dynamic formulation, the following control law is applied to the master manipulator

$$\mathbf{F}_m = \mathbf{\Lambda}_m(\mathbf{q}_m)\lambda\dot{\mathbf{x}}_m + \bar{\mathbf{F}}_m, \tag{2.115}$$

while the controller for the slave robot is an adaptive scheme given by (2.82).

The coordinating torques, $\bar{\tau}_i$, are chosen as

$$\begin{aligned} \bar{\mathbf{F}}_m &= \mathbf{K}(\mathbf{r}_m - \mathbf{r}_s(t-T)) \\ \bar{\tau}_s &= \mathbf{K}\mathbf{\Phi}_s(\mathbf{q}_s)^{-1}(\mathbf{r}_s - \mathbf{r}_m(t-T)) \end{aligned} \tag{2.116}$$

where $\mathbf{K} = \mathbf{K}^T > 0$ is a constant matrix; $\mathbf{q}_m$ and $\mathbf{q}_s$ are the configurations of the master and slave robot respectively. The interconnection variables, $\mathbf{r}_i$, are defined as $\mathbf{r}_i = \dot{\mathbf{x}}_i + \lambda \mathbf{x}_i$, λ is a positive constant matrix, and $\bar{\mathbf{J}}(\mathbf{q}) = \mathbf{J}(\mathbf{q})\mathbf{M}(\mathbf{q})$ is a decoupling matrix which maps between the independent joints space, u, and Cartesian output space, x, satisfying

$$\dot{\mathbf{x}} = \bar{\mathbf{J}}(\mathbf{q})\mathbf{u} \qquad \text{and} \qquad \mathbf{u} = \bar{\mathbf{J}}(\mathbf{q})^{-1}\dot{\mathbf{x}},$$

where $\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}$. The controller was implemented on a simulated system, using a Phantom Omni to control a wheeled mobile robot. Position synchronization was achieved, despite initial position offset between the master and slave system. In addition, the system remained stable when virtual rigid obstacles were introduced.

Tai and Murakami (2009) [115] use a control index called *motion initiative* (MI) in the same way as the S_{mode} variable, proposed by Farkhatdinov, Ryu and Poduraev (2008)[61]. Depending on the distance to the target, the controller is set in different phases. A multivariable controller based on inverse dynamics [92] is used to control the acceleration, rather than the torque. The position of the slave system is denoted $\mathbf{x}_s$ and is related to the generalized coordinates $\mathbf{q}$ by a Jacobian matrix

$$\begin{aligned} \dot{\mathbf{x}}_s &= \mathbf{J}_s\dot{\mathbf{q}}_s \\ \ddot{\mathbf{q}}_s &= \mathbf{J}_{sw}^{\dagger}\ddot{\mathbf{x}}_s + (\mathbf{I} - \mathbf{J}_{sw}^{\dagger}\mathbf{J}_s)\eta, \end{aligned} \tag{2.117}$$

where η is an arbitrary null space vector. $\mathbf{J}_{sw}^{\dagger}$ is the weighted pseudo inverse matrix of the Jacobian $\mathbf{J}$, and defined as

$$\mathbf{J}_{sw}^{\dagger} = \mathbf{W}^{-1}\mathbf{J}_s^T(\mathbf{J}_s\mathbf{W}^{-1}\mathbf{J}_s^T)^{-1}, \tag{2.118}$$

with

$$\mathbf{W} = \text{diag}\{\mathbf{w}_p \quad \mathbf{w}_m\}, \tag{2.119}$$

where $\mathbf{w}_p$ and $\mathbf{w}_m$ are functions of the *MI*-variable. The controllers for the master and slave are given as

$$\begin{aligned} \ddot{\mathbf{x}}_m &= C_p(\mathbf{x}_s - \mathbf{x}_m) + C_f(f_m + f_s) \\ \ddot{\mathbf{x}}_s &= C_p(\mathbf{x}_m - \mathbf{x}_s) + C_f(f_m + f_s). \end{aligned} \tag{2.120}$$

The variables C_p and C_f are position gain and force gain, respectively. The generalized forces are given as

$$\begin{aligned} \tau_m &= \mathbf{D}_m\ddot{\mathbf{q}}_m + \mathbf{h}_m(\mathbf{q}_m, \dot{\mathbf{q}}_m) \\ \tau_s &= \mathbf{D}_s\ddot{\mathbf{q}}_s + \mathbf{h}_s(\mathbf{q}_s, \dot{\mathbf{q}}_s), \end{aligned} \tag{2.121}$$

where $\mathbf{D}_i$ are the inertial matrices and $\mathbf{h}_i$ are given as

$$\mathbf{h}_i = \mathbf{C}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\dot{\mathbf{q}}_i + \mathbf{g}_i(\mathbf{q}). \tag{2.122}$$

$\mathbf{C}_i$ are the Coriolis and centrifugal effects, and $\mathbf{g}_i$ are the gravitational term. Time delay is not handled in this paper.

To implement a controller on a computer, it is convenient to use an application programming interface (API). One such API is Robot Operating System, and described in the next section.

2.6 Robot Operating System (ROS)

ROS, or Robot Operating System, is an open-source, high-level API, and a so called meta-operating system for robotics applications. Being a meta-operating system means that ROS provides operating system-like services, but has to run on another operating system. ROS is primarily tested on Ubuntu and Mac OS x, though there exists experimental versions for other operating systems, such as Fedora, Gentoo, Arch and Windows. Even though ROS supports libraries written in several programming languages, the main supported libraries are written in either C++ or Python [44]. The graphical representation is specified with Unified Robot Description Format (URDF), an Xml format for representing robot model.

A larger ROS program often consists of several *nodes*, which are processes performing computation. The communication between the nodes involves sending and receiving *messages*, where a message corresponds to a data structure. The different communication styles provided for passing messages, includes synchronous communication, using *services*, and asynchronous streaming, using *topics*. When using topics, the node which generates the data is said to publish, and the node which reads is said to subscribe.

An important tool in ROS is *tf*. tf makes it possible to keep track of multiple coordinate frames over time, where the frames are organized in a tree structure, buffered in time [116].

The purpose of the ROS program used in this study is to implement a control architecture, calculating the desired control input to the mobile manipulator and the force feedback to the human operator. This chapter presents several elements important to design a controller, with the most important parts summarized in the next section.

2.7 Summary

The main challenge in bilateral teleoperation is that the reference position for a slave manipulator, given by a master manipulator/human operated joystick, is affected by the last measured position of the slave manipulator. By introducing time delay, the system, given by the master and slave manipulator connected through a communication network, can exhibit large oscillations.

Not only is time delay challenging, but the use of mobile manipulator introduce several aspects which differ from standard manipulators. Mobile manipulators often have an increased workspace, in addition to being more redundant, and having nonholonomic constraints. Increased workspace and redundancy allow mobile manipulators to solve additional tasks, and some tasks in more ways than manipulators with fixed base. Though, redundancy and nonholonomic constraints require a more intelligent controller than otherwise.

Since the master and slave manipulator is non-identical, it would be beneficial to use the operational space formulation. By describing the end-effector equations of motion in operational space, it is possible to specify a command vector for the end-effector, neglecting the differences in design. It is also suggested to use the operational space formulation to impose several tasks simultaneously, but with different priority, for instance first comply with the constraints, then ensuring a given position for the end-effector before maintaining a preferred posture.

Manipulability measure is a function which quantifies the ability of an end-effector to move in different directions. For a mobile manipulator, the optimal manipulability measure depends on how active the movable base is when moving the arm.

Scattering theory is one of the major breakthroughs in handling time delay in teleoperation. By considering the force and velocity as voltage and current, a relationship between passivity and the scattering operator was found.

This was later used to derive what is called scattering variables, that are combinations of the force and velocity from the master and slave manipulator, respectively. These results formed the basis for many of the earliest controllers, and are still important tools in teleoperation.

To handle time delay, different controllers have been proposed. The controllers presented in Section 2.5.2 have different properties, but are mainly focusing on identical systems. To make it easier to compare, the different schemes are listed in Table 2.7. The table shows whether the scheme is passive with constant and variable time-delay, if the position is tracked and if the scheme is based on scattering variables. As seen in the table, all the controllers are stable for constant time-delay, but only five of them are stable for variable time-delay. Of the different listed controllers, *classic scattering scheme*, *symmetric impedance matching*, *classic scattering for variable time-delays* and *passive output interconnection* can not guarantee position tracking. The main difference between the two groups are the use of scattering variable.

Scheme	Time-delays		Pos. Track.	Scatt.based
	Const.	Var.		
Classical scattering	√			√
Symm. Imp. Matching	√			√
Pos. Track. Controller	√		√	√
Symm. Pos. Tracking	√		√	√
Classical Scatt. Var. T.-D.	√	√		√
Pos. Track. Var. T.-D.	√	√	√	√
PD + d	√	√	√	
P + d	√	√	√	
Passive Output Intercon.	√	√		
Asymp.regulation	√		√	
Scatt.State Synch.	√	√	√	√

Table 2.2: List of variables and parameters [96].

Force rendering scheme is a way to model a virtual environmental force. The purpose is to give the human operator information about the surrounding areas through the haptic joystick.

Various combinations of operational space formulation, manipulability measure, scattering theory, delay handling, and force rendering are used to make stable, intuitive and informative controllers for teleoperation of mobile manipulators.

The values used in the model for the system are unique for each problem, and have to be derived. The topic of the next chapter is the derivation of the kinematics for the master and slave manipulator, that is, the relationship between the generalized coordinates and the position of the end-effectors.

Chapter 3

Master and Slave Kinematics

Kinematics are often used to describe the relationship between generalized and Cartesian coordinates. In this study, the master and slave manipulator are kinematically and dynamically different, which means that a joint configuration of the human operated joystick can not be used as a reference value for the slave robot directly, and vice versa. Some obvious differences are the number of joints and fixed versus movable base. It is therefore necessary to use operational space coordinates to control the manipulators.

The first section describes forward kinematics, used to find the position and orientation of the different joints in Cartesian coordinates. The transformation from joint velocities to velocities in operational space is covered by Section 3.2. Section 3.3 describes how to find the desired velocity in joint space, based on a reference velocity in Cartesian coordinates. Section 3.4 and 3.5 derives the kinematics specifically for the master and slave manipulator, before Section 3.6 summarizes the key elements of this chapter.

3.1 Forward Kinematics

The first step in describing the kinematics, is finding the forward kinematics. The forward kinematics describe the position and orientation of a frame, often fixed to a link, relative to a previous frame. The main purpose of forward kinematics is to determine the position and orientation of the end-effector, or other links, in Cartesian coordinates, based on the configuration of the joints.

The position and orientation of the end-effector in Cartesian coordinates, ξ, is given by the generalized coordinates, $\mathbf{q}$, through the function $\mathbf{f}$, and can be written as

$$\xi = \mathbf{f}(\mathbf{q}). \tag{3.1}$$

The function, $\mathbf{f}$, can be found using a transformation matrix, which is an importing tool in this analysis, and will be the next topic.

3.1.1 Transformation Matrix

A common way of representing the forward kinematics is by using a transformation matrix, usually denoted by T. A transformation matrix from frame j to frame i consists of a set of homogeneous transformations, and can be written as

$$\begin{aligned} T_j^i &= A_{j+1}A_{j+2}...A_{i-1}A_i = T_j^{i-1}A_i \quad \text{if } j < i \\ T_j^i &= I_4 \quad \text{if } j = i \\ T_j^i &= (T_i^j)^{-1} \quad \text{if } j > i, \end{aligned}$$

where I_4 is an order 4 identity matrix. The homogeneous transformation A_i transforms the coordinates of a point from frame i to frame $i-1$. The matrix A_i is a function of only a single joint variable, q_i, and is of the form

$$A_i(q_i) = \begin{bmatrix} R_{i-1}^i & \mathbf{p}_{i-1}^i \\ 0 & 1 \end{bmatrix}, \tag{3.2}$$

giving

$$T_j^i = \begin{bmatrix} R_j^i & \mathbf{p}_j^i \\ 0 & 1 \end{bmatrix}. \tag{3.3}$$

The matrix R_j^i describes the orientation of frame i relative to frame j, while $\mathbf{p}_j^i$ is the position of the origin of frame i, described in frame j.

To find the homogeneous transformations A_i, the Denavit-Hartenberg convention is used. The next topic includes therefore a short description of this convention.

3.1.2 Denavit-Hartenberg Representation

A commonly used convention for selecting frames of reference and finding the transformation matrices, is the Denavit-Hartenberg, or D-H, convention.

In the D-H convention, A_i is represented as a product of four "basic" transformations

$$A_i = A_i(q_i) = Rot_{z,\theta_i} Trans_{z,d_i} Trans_{x,a_i} Rot_{x,\alpha_i}. \tag{3.4}$$

The angle α_i is the angle between the axis $\mathbf{z}_{i-1}$ and $\mathbf{z}_i$, and is measured as a rotation about $\mathbf{x}_i$. The parameter a_i is the distance between the axes $\mathbf{z}_{i-1}$ and $\mathbf{z}_i$, measured along $\mathbf{x}_i$. The parameter d_i is the distance between the origin $\mathbf{o}_{i-1}$ and the intersection of the $\mathbf{x}_i$ axis, measured along the $\mathbf{z}_{i-1}$ axis. Finally, θ_i is the angle between the $\mathbf{x}_{i-1}$ and $\mathbf{x}_i$ axis, measured as a rotation about $\mathbf{z}_{i-1}$. A more detailed algorithm for using DH-convention is described in Section B.1.

The homogeneous transformation matrices A_i, found using the D-H convention, are then used finding the transformation matrices T_j^i. In particular T_g^i, which is the transformation from frame i to the global frame, is defined as

$$T_g^i = A_0 A_1 ... A_i, \tag{3.5}$$

where A_0 is a fixed transformation from frame 0 to the global frame, and do not depend on a generalized coordinate. A fixed transformation is used when z_0, defined by the direction of the generalized coordinate q_1, do not point in the direction of the preferred global frame.

While the position and orientation of a frame, given by the configuration of the joints, are found through forward kinematics, the connection from joint velocity to instantaneous velocity is described with forward differential kinematics, which will be discussed in the next section.

3.2 Forward Differential Kinematics

The forward differential kinematics is the transformation from the generalized velocity, $\dot{\mathbf{q}}$, to the instantaneous velocity of the end-effector, $\dot{\xi}$, given in Cartesian coordinates. Each entry of the instantaneous velocity, $\dot{\xi}_i$, is linearly dependent on each generalized velocities, $\dot{q}_j$, and can be written as

$$\dot{\xi} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \tag{3.6}$$

where $\mathbf{J}$ is the Jacobian. The Jacobian is found as the partial derivative of the model for the forward kinematics (3.1) with respect to the joint velocity. The forward kinematics model is based on the transformation matrices described in Section 3.1.

For nonholonomic systems, some (or all) of the generalized velocities are not directly controllable, but depend on the control input $\mathbf{u}$ through the configuration differential kinematic model $\mathbf{M}(\mathbf{q})$, written as

$$\dot{\mathbf{q}} = \mathbf{M}(\mathbf{q})\mathbf{u}. \tag{3.7}$$

Hence,

$$\dot{\xi} = \mathbf{J}(\mathbf{q})\mathbf{M}(\mathbf{q})\mathbf{u} = \bar{\mathbf{J}}(\mathbf{q})\mathbf{u}, \tag{3.8}$$

for nonholonomic systems. Here, the matrix $\bar{\mathbf{J}}$ is defined as $\bar{\mathbf{J}}(\mathbf{q}) = \mathbf{J}(\mathbf{q})\mathbf{M}(\mathbf{q})$.

In spite of the nonholonomic constraints, the calculation of the Jacobian, $\mathbf{J}$, is similar to a holonomic system.

In the general case the velocity $\dot{\xi}$ consists of a linear velocity $\mathbf{v}$ and an angular velocity ω, up to a total of six dimensions. The Jacobian can then be partitioned to a linear and an angular part, written as

$$\dot{\xi} = \begin{bmatrix} \mathbf{v} \\ \omega \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{\mathbf{v}} \\ \mathbf{J}_{\omega} \end{bmatrix} \dot{\mathbf{q}}. \tag{3.9}$$

The procedure for finding $\mathbf{J}_{\mathbf{v}}$ and $\mathbf{J}_{\omega}$ deviates slightly, and the following sections will derive each of the Jacobians before they are combined.

3.2.1 Jacobian for Linear Velocity

For a robot with n generalized coordinates, the linear velocity of the end-effector relative to the global frame, $\mathbf{v}_g^n$, is defined as the derivative of the position, $\mathbf{p}_g^n$, giving

$$\mathbf{v}_g^n = \sum_{i=1}^{n} \frac{\partial \mathbf{p}_g^n}{\partial q_i} \dot{q}_i. \tag{3.10}$$

The end-effector position, $\mathbf{p}_g^n$, can be written as

$$\mathbf{p}_g^n = \mathbf{p}_g^{i-1} + R_g^{i-1}\mathbf{p}_{i-1}^n = \mathbf{p}_g^{i-1} + R_g^{i-1}\mathbf{p}_{i-1}^i + R_g^i\mathbf{p}_i^n, \tag{3.11}$$

and if only joint i moves, both R_g^{i-1}, $\mathbf{p}_g^{i-1}$ and $\mathbf{p}_i^n$ are constant.

If joint i is revolute and the position of frame i in global coordinates, $\mathbf{p}_g^i$, is written as $\mathbf{o}_i$, it is possible to write

$$\mathbf{o}_n - \mathbf{o}_{i-1} = R_g^{i-1}\mathbf{p}_{i-1}^n. \tag{3.12}$$

Since R_g^{i-1}, $\mathbf{p}_g^{i-1}$ and $\mathbf{p}_i^n$ are constant, the end-effector velocity is given as

$$\dot{\mathbf{p}}_g^n = R_g^{i-1}\dot{\mathbf{p}}_{i-1}^n. \tag{3.13}$$

The generalized coordinate, $q_i = \theta_i$, is a rotation about $\mathbf{z}_i$, such that

$$\dot{\mathbf{p}}_{i-1}^n = \dot{q}_i \mathbf{k} \times \mathbf{p}_{i-1}^n. \tag{3.14}$$

Inserting the expression for the end-effector velocity in frame $i-1$, given by (3.14), into the expression for the end-effector velocity in global coordinates (3.13), gives

$$\begin{aligned} \dot{\mathbf{p}}_g^n &= R_g^{i-1}(\dot{q}_i \mathbf{k} \times \mathbf{p}_{i-1}^n) \\ &= \dot{q}_i R_g^{i-1}\mathbf{k} \times R_g^{i-1}\mathbf{p}_{i-1}^n \\ &= \dot{q}_i \mathbf{z}_{i-1} \times (\mathbf{o}_n - \mathbf{o}_{i-1}). \end{aligned} \tag{3.15}$$

Hence,

$$\frac{\partial \mathbf{p}_g^n}{\partial q_i} = \mathbf{z}_{i-1} \times (\mathbf{o}_n - \mathbf{o}_{i-1}) \tag{3.16}$$

for revolute joints.

For prismatic joints, the generalized coordinate, $q_i = d_i$, corresponds to a displacement along $\mathbf{z}_{i-1}$, while the rotation matrix R_g^i is constant. This leads to the following expression for the velocity

$$\dot{\mathbf{p}}_g^n = R_g^{i-1}\dot{\mathbf{p}}_{i-1}^i = R_g^{i-1}\dot{q}_i \mathbf{k}. \tag{3.17}$$

Hence,

$$\frac{\partial \mathbf{p}_g^n}{\partial q_i} = R_g^{i-1}\mathbf{k} = \mathbf{z}_{i-1} \tag{3.18}$$

if joint i is prismatic.

To summarize, the Jacobian for the linear velocity, $\mathbf{J_v}$, is given as

$$\mathbf{J_v} = [\mathbf{J}_{\mathrm{v}1} ... \mathbf{J}_{\mathrm{v}n}], \tag{3.19}$$

where the i-th column, $\mathbf{J}_{\mathrm{v}i}$, can be written as

$$\mathbf{J}_{\mathrm{v}i} = \mathbf{z}_{i-1} \tag{3.20}$$

if joint i is prismatic and

$$\mathbf{J}_{vi} = \mathbf{z}_{i-1} \times (\mathbf{o}_n - \mathbf{o}_{i-1}) \tag{3.21}$$

if joint i is revolute.

The same approach can be used to find the instantaneous linear velocity for other links, and relative to other frames. The next step is to find the Jacobian for the angular velocity.

3.2.2 Jacobian for Angular Velocity

The expression for the angular velocity of frame i, relative to frame $i-1$, is given by

$$\omega_{i-1}^{i} = \dot{\theta}_i \mathbf{k}. \tag{3.22}$$

If joint i is revolute, the angle θ_i corresponds to the joint variable q_i and the angular velocity, ω_{i-1}^{i}, is given as

$$\omega_{i-1}^{i} = \dot{q}_i \mathbf{k}. \tag{3.23}$$

If joint i is prismatic, the angle θ_i is constant and the angular velocity, ω_{i-1}^{i}, is zero. Thus, the angular velocity of the end-effector in the global frame, ω_n, can be written as

$$\omega_n = \rho_1 \dot{q}_1 \mathbf{k} + \rho_2 \dot{q}_2 R_g^1 \mathbf{k} + ... + \rho_n \dot{q}_n R_g^{n-1} \mathbf{k} = \sum_{i=1}^{n} \rho_i \dot{q}_i \mathbf{z}_{i-1}, \tag{3.24}$$

where $\mathbf{z}_{i-1} = R_g^{i-1}\mathbf{k}$, and ρ_i is 1 if joint i is revolute and 0 if the joint is prismatic. Thus the i-th column of the Jacobian for the angular velocity, $\mathbf{J}_\omega$, can be written as

$$\mathbf{J}_{\omega i} = 0 \tag{3.25}$$

if joint i is prismatic and

$$\mathbf{J}_{\omega i} = \mathbf{z}_{i-1} \tag{3.26}$$

if joint i is revolute.

The angular velocity is rarely found for other links than the end-effector, since the angle of a link in the global frame seldom leads to a violation of any constraints itself, but it is still possible to find the instantaneous angular velocity for each link, given relative to different frames.

To find the entire instantaneous velocity, it is necessary to combine the two Jacobians.

3.2.3 Combined Jacobian

The i-th column of the Jacobian is found by combining the expressions for the upper and lower part of the column, $\mathbf{J}_{\mathrm{v}i}$ and $\mathbf{J}_{\omega i}$. If joint i is prismatic, $\mathbf{J}_i$ is given by (3.20) and (3.25) as

$$\mathbf{J}_i = \begin{bmatrix} \mathbf{z}_{i-1} \\ 0 \end{bmatrix}. \tag{3.27}$$

If joint i is revolute, the i-th column of the Jacobian is given by (3.21) and (3.26) as

$$\mathbf{J}_i = \begin{bmatrix} \mathbf{z}_{i-1} \times (\mathbf{o}_n - \mathbf{o}_{i-1}) \\ \mathbf{z}_{i-1} \end{bmatrix}. \tag{3.28}$$

The Jacobian is initially used as a transformation from the velocity in joint space to linear and angular velocity in the operational space, but the Jacobian is also important in finding the desired joint velocity based on a desired linear and angular velocity in operational space. This connection is called inverse differential kinematics, and is the next topic.

3.3 Inverse Differential Kinematics

Inverse differential kinematics address the problem of finding a set of joint velocities, which realizes a desired linear and angular end-effector velocity. For nonholonomic systems, the problem is finding the desired control input, $\mathbf{u}^*$, based on the desired end-effector velocity. Since the forward differential kinematics is given by (3.6), or by (3.8) for nonholonomic systems, the solution for the inverse differential kinematics is ideally given as

$$\dot{\mathbf{q}}^* = \mathbf{J}(\mathbf{q})^{-1}\dot{\xi}^*, \tag{3.29}$$

where $\dot{\mathbf{q}}^*$ and $\dot{\xi}^*$ are the desired joint and Cartesian velocities, respectively. Or

$$\mathbf{u}^* = \bar{\mathbf{J}}(\mathbf{q})^{-1}\dot{\xi}^*, \tag{3.30}$$

for nonholonomic systems. But the Jacobian is not always invertible and $\bar{\mathbf{J}}$ is never invertible, which means that finding $\dot{\mathbf{q}}^*$ or $\mathbf{u}^*$ is not trivial.

3.3.1 Inverse Jacobian

The inverse of the Jacobian in (3.29) can only be found if the Jacobian, $\mathbf{J}$, is square and has full rank. In most applications the Jacobian is either not square or the system encounter situations where the rank drops, so that an alternative solution must be found.

If the Jacobian is not square, it has either more rows than columns, or vice versa. For a system where the Jacobian has more rows than columns, it is not possible to realize all the velocities independently in operational space. On the other hand, a system where the Jacobian has more columns than rows is said to be redundant, which means that the same velocity in operational space can be realized by a numerous (or infinite) combination of joint velocities. In both cases it does not exists an exact solution for every desired end-effector velocity.

For systems with a square, full rank Jacobian, each desired end-effector velocity has a corresponding desired joint velocity. However, if a joint is fully extended, the Jacobian is said to become singular and not invertible because the rank drops. This happens because the instantaneous velocity of the end-effector can no longer be fulfilled independently in all the previous dimensions. As a result, the instantaneous velocity of the end-effector in some direction has several solutions.

Pseudoinverse

One way of finding the inverse differential kinematics is to use the pseudoinverse of the Jacobian. The most commonly used type of pseudoinverse is the Moore-Penrose pseudoinverse. A Moore-Penrose pseudoinverse (from now on just called pseudoinverse) of $\mathbf{J}$ is defined as a matrix $\mathbf{J}^\dagger$ which satisfies the following criteria:

1. $\mathbf{J}\mathbf{J}^\dagger\mathbf{J} = \mathbf{J}$
2. $\mathbf{J}^\dagger\mathbf{J}\mathbf{J}^\dagger = \mathbf{J}^\dagger$
3. $(\mathbf{J}\mathbf{J}^\dagger)^* = \mathbf{J}\mathbf{J}^\dagger$
4. $(\mathbf{J}^\dagger\mathbf{J})^* = \mathbf{J}^\dagger\mathbf{J}$

For a real matrix $\mathbf{J}$, the pseudoinverse can either be found as the limit

$$\mathbf{J}^\dagger = \lim_{\delta \to 0^+} \mathbf{J}^T(\mathbf{J}\mathbf{J}^T + \delta\mathbf{I})^{-1}, \tag{3.31}$$

or by using singular value decomposition (SVD). A more detailed description of computing the SVD can be found in Section B.2.

According to Nakamura (1991) [65], all the solutions of inverse differential kinematics are given by

$$\dot{\mathbf{q}}^* = \mathbf{J}(\mathbf{q})^\dagger\dot{\xi}^* + (\mathbf{I}_n - \mathbf{J}(\mathbf{q})^\dagger\mathbf{J}(\mathbf{q}))\mathbf{w}, \tag{3.32}$$

for an arbitrary vector $\mathbf{w}$ of dimension n, where $\mathbf{N}(\mathbf{q}) = \mathbf{I}_n - \mathbf{J}^\dagger(\mathbf{q})\mathbf{J}(\mathbf{q})$ spans the null space of $\mathbf{J}$. The solutions are equal to solving the following optimization problem

$$\min \|\mathbf{J}(\mathbf{q})\dot{\mathbf{q}}^* - \dot{\xi}^*\|, \tag{3.33}$$

where the pseudoinverse solution is found by choosing $\mathbf{w} = \mathbf{0}$.

Similarly, the desired control input $\mathbf{u}^*$ for a nonholonomic system, given by (3.8), can be found as

$$\mathbf{u}^* = \bar{\mathbf{J}}^\dagger(\mathbf{q})\dot{\xi}^* + (\mathbf{I}_n - \bar{\mathbf{J}}^\dagger(\mathbf{q})\bar{\mathbf{J}}(\mathbf{q}))\mathbf{w}. \tag{3.34}$$

Since the vector $\mathbf{w}$ operates in the null space of $\mathbf{J}(\mathbf{q})$ or $\bar{\mathbf{J}}(\mathbf{q})$, it is possible to choose an additional task that do not affect the end-effector velocity.

Extended Jacobian

An alternative to the use of pseudoinverse is the extended Jacobian technique, proposed by Baillieul (1985) [117] and Chang (1987) [118]. The idea is to enforce a number of additional tasks along with the original end-effector task [119].

Consider a system where the end-effector location consists of m operational coordinates, $\xi = [\xi_1\ \xi_2\ ...\ \xi_m]$, given by (3.1), with n generalized coordinates $\mathbf{q} = [q_1\ q_2\ ...\ q_n]$. If the Jacobian has rank m then the null space of the Jacobian, $\mathbf{N}(\mathbf{q})$, has rank $n - m$. The set of $n - m$ additional tasks can be written in vector form as

$$\mathbf{h}(\mathbf{q}) = \mathbf{0}, \tag{3.35}$$

with the instantaneous velocity given as $\frac{\partial \mathbf{h}(\mathbf{q})}{\partial \mathbf{q}}\dot{\mathbf{q}} = \mathbf{0}$. Combining the two objectives, to meet the additional task while the end-effector has the desired linear and angular velocity, gives

$$\begin{bmatrix} \mathbf{J}(\mathbf{q}) \\ \frac{\partial \mathbf{h}(\mathbf{q})}{\partial \mathbf{q}} \end{bmatrix} \dot{\mathbf{q}} = \mathbf{J}_{\text{ext}}(\mathbf{q})\dot{\mathbf{q}} = \begin{bmatrix} \dot{\xi} \\ \mathbf{0} \end{bmatrix}, \tag{3.36}$$

where $\mathbf{J}_{\text{ext}}$ is square and called the extended Jacobian. As long as $\mathbf{J}_{\text{ext}}$ do not get singular, the desired generalized velocity, $\dot{\mathbf{q}}^*$, can be found as

$$\dot{\mathbf{q}}^* = \mathbf{J}_{\text{ext}}(\mathbf{q})^{-1} \begin{bmatrix} \dot{\xi}^* \\ \mathbf{0} \end{bmatrix}, \tag{3.37}$$

where $\dot{\xi}^*$ is the desired linear and angular end-effector velocity.

Both with pseudoinverse and extended Jacobian, various additional tasks can be specified for redundant manipulators. The next topic will cover how to impose the tasks, described as velocity in either operational space or joint space.

3.3.2 Velocity Tasks, Desired Velocity in Operational Space

For a redundant manipulator, a robot with more controllable inputs than the DOF for the end-effector configuration, it is possible to impose one or more tasks in addition to a desired end-effector velocity. A task i is denoted $\mathbf{t}_i$, and can be a desired behavior in either Cartesian or generalized coordinates. An additional task may be to avoid joint limitation, comply with movement restrictions, or optimize the pose or manipulability of the manipulator. The same tasks can also be specified as velocities, defined as velocity tasks and denoted $\dot{\mathbf{t}}_i$

If the approach using the pseudoinverse is used, the tasks are organized in a hierarchy, that is, every task has a different priority, and a task $\mathbf{t}_i$ do not affect tasks with higher priorities. For a system with n generalized coordinates and μ tasks, the desired generalized velocity is

$$\dot{\mathbf{q}}^* = \mathbf{J}_1^\dagger(\mathbf{q})\dot{\mathbf{t}}_1 + \mathbf{N}_1(\mathbf{J}_2^\dagger(\mathbf{q})\dot{\mathbf{t}}_2 + \mathbf{N}_2(... + \mathbf{J}_\mu^\dagger(\mathbf{q})\dot{\mathbf{t}}_\mu)), \tag{3.38}$$

where $\mathbf{N}_i(\mathbf{q}) = \mathbf{I}_n - \mathbf{J}_i^\dagger(\mathbf{q})\mathbf{J}_i(\mathbf{q})$, $\mathbf{J}_i = \frac{\partial \mathbf{t}_i}{\partial \mathbf{q}}$ and $\dot{\mathbf{t}}_i = \mathbf{J}_i\dot{\mathbf{q}}$. A similar hierarchy

can be made for nonholonomic systems with ν control inputs,

$$\mathbf{u}^* = \bar{\mathbf{J}}_1^\dagger(\mathbf{q})\dot{\mathbf{t}}_1 + \mathbf{N}_1(\bar{\mathbf{J}}_2^\dagger(\mathbf{q})\dot{\mathbf{t}}_2 + \mathbf{N}_2(... + \bar{\mathbf{J}}_\mu^\dagger(\mathbf{q})\dot{\mathbf{t}}_\mu)), \tag{3.39}$$

with $\mathbf{N}_i(\mathbf{q}) = \mathbf{I}_\nu - \bar{\mathbf{J}}_i^\dagger(\mathbf{q})\bar{\mathbf{J}}_i(\mathbf{q})$, $\bar{\mathbf{J}}_i(\mathbf{q}) = \mathbf{J}_i(\mathbf{q})\mathbf{M}(\mathbf{q})$ and $\dot{\mathbf{t}}_i = \bar{\mathbf{J}}_i\mathbf{u}$. Here $\mathbf{J}_i$ is the Jacobian for task i and $\mathbf{M}$ is the configuration differential kinematic model, which maps the control input $\mathbf{u}$ to the generalized velocity $\dot{\mathbf{q}}$.

The main benefit with this approach is the low computation complexity. Each new, lower priority task is added in the null space of the earlier tasks without calculating a new solution for the higher priority tasks. There is no bound on the number of tasks which can be imposed and it is not necessary to check if all the tasks can be solved simultaneously, because each task is guaranteed to be solved according to the hierarchy. In addition, it will always exist a solution.

The problem with this approach is that two tasks can not have the same priority. For example, it is not possible to guarantee that a restriction is met and, in the same time, be sure that the end-effector velocity is not affected, even though it exists a generalized velocity that satisfies both tasks.

To solve the problems stated above, it is possible to use the extended Jacobian, but then none of the aforementioned benefits will longer apply. That is, for each new task the extended Jacobian, $\mathbf{J}_{\text{ext}}$, has to be calculated, the total number of dimensions for the tasks can not exceed the number of generalized coordinates, and it does not exist a solution if $\mathbf{J}_{\text{ext}}$ becomes singular.

The theory presented in the first part of this chapter is used on a specific master and slave manipulator. Next, a short description of the kinematics of the master manipulator is presented.

3.4 Master Manipulator Kinematics

The human operated joystick is often referred to as master (or local) manipulator, and is used to send a command signal to a controlled robot based on the movement enforced by the human operator. In this thesis a Phantom Omni is used, a haptic joystick (see Section 2.4.1) which can measure movement in 6 DOF and give force feedback in 3 DOF.

In the following section, the kinematics for the Phantom Omni are derived. The main purpose of the kinematics is to relate the measured output from the joystick to a reference in Cartesian coordinates, and to apply force feedback in the desired direction, based on calculations made in operational space.

3.4.1 Forward Kinematics

As previously mentioned, the Phantom Omni can measure the configuration of the joystick in 6 dimensions, whereof three coordinates describing the position and three describing the rotation. The position of the joystick is

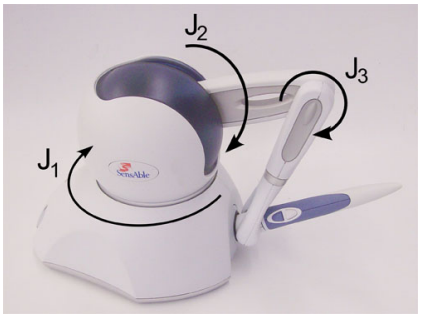


Figure 3.1: Phantom Omni, joint angles [120].

measured at the attachment point of the pen, and is given in a global joystick frame, where the z-axis is defined as the line from the base tower towards the pen in initial position (seen in Figure 3.1), the y-axis points upwards, perpendicular to the surface, and the x-axis completes the coordinate system. The position (0, 0, 0) is located at the third joint, when the first link is horizontal. In the initial position the angles J_1, J_2 and J_3 in Figure 3.1

are (0, 0, 0), and the position is given as $(0, L_3 - L_2, -L_4 + L_1)$. Here $L_1 = L_2 = 0.135$m are the length of the links, and the following variables defined as $A = 0.035$m, $L_3 = 0.025$m and $L_4 = L_1 + A$. While J_1 and J_2 are the angles of joint 1 and 2 respectively, J_3 is the angle between a fixed axis, given in global coordinates, and link 3.

Seen from the top, the relationship between J_1 and the x- and z-axis is shown in Figure 3.2. Figure 3.3 shows the kinematic chain from the side, and the relationship between the distance from the base tower, R, the hight over the second joint, Y, and the angles J_2 and J_3. $J_{2,0} = 0.15$ and $J_{3,0} = -0.25$ are the initial angles relative to the horizontal and vertical plane, respectively.

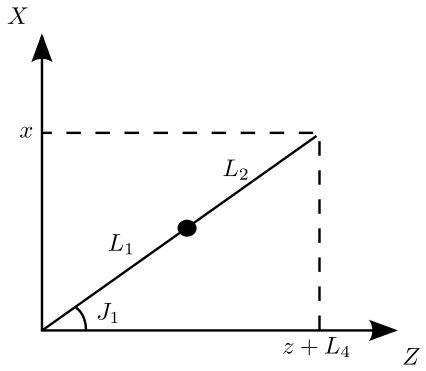


Figure 3.2: The kinematics of Phantom Omni seen from the top.

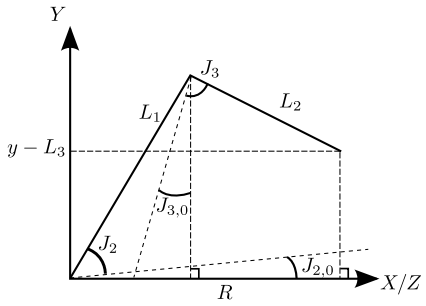


Figure 3.3: The kinematics of Phantom Omni seen from the side.

The position of the joystick is calculated as

$$\begin{aligned} x &= -\sin J_1(L_1\cos(J_2+J_{2,0})+L_2\sin(J_3+J_{3,0})) \\ y &= L_3+L_1\sin(J_2+J_{2,0})-L_2\cos(J_3+J_{3,0}) \\ z &= -L_4+\cos J_1(L_1\cos(J_2+J_{2,0})+L_2\sin(J_3+J_{3,0})). \end{aligned} \tag{3.40}$$

For the pen, the orientation relative to the second link is given by the three angles J_4, J_5 and J_6 , shown in Figure 3.4. The last three angles all operates around $-\pi$. By using Euler angles, the orientation of the pen relative to the global frame of the joystick can be calculated as six subsequent rotations

$$R = R_{\pi,z}R_{J_1,y}R_{\tilde{J}_3,x}R_{-J_4,y}R_{\pi+J_5,x}R_{J_6+\pi/2,z}, \tag{3.41}$$

where $\tilde{J}_3 = J_3 + J_{3,0}$. In this thesis, J_6 is only used as a reference for the rotation of the last joint of the slave manipulator, and do not affect the orientation of the joystick frame. Hence, the orientation of the joystick frame is given as

$$R = R_1R_2...R_6 = R_{\pi,z}R_{J_1,y}R_{\tilde{J}_3,x}R_{-J_4,y}R_{\pi+J_5,x}R_{-\pi/2,z}, \tag{3.42}$$

where the values of the matrices can be found in Section B.3.

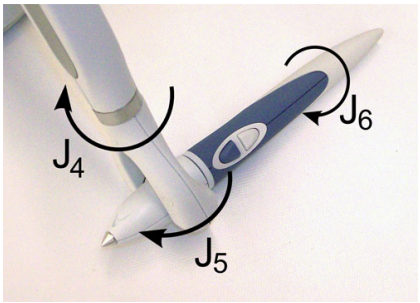


Figure 3.4: Phantom Omni, pen angles [120].

The velocity of the joystick is measured as the time derivative of the position in operational space, also, the feedback force is specified in Cartesian coordinates and converted by a provided library to the appropriated torques. It is

therefore not necessary to find the inverse kinematics or forward differential kinematics for the joystick. A more extensive derivation of the kinematics for the Phantom Omni is found in Silva et al. (2009) [121], though the initial values $J_{2,0}$ and $J_{3,0}$ have been disregarded in this thesis.

Next, the kinematics for the slave manipulator is derived.

3.5 Slave Manipulator Kinematics

The slave manipulator in this study is a mobile manipulator (see Section 2.3), and consists of a 7-link robot arm mounted on a four wheeled mobile base. The manipulator is an LWA3 from Schunk, shown in Figure 3.5, while the base is a Seekur Jr. from Adept MobileRobots, shown in Figure 3.6.

Figure 3.5: LWA3, manipulator from Schunk GmbH & Co. KG, Germany [122].

Figure 3.6: SeekurJr, mobile base from Adept MobileRobots, USA [60].

3.5.1 Forward Kinematics

With a mobile base, the area where the slave manipulator is able to move is in theory unlimited, which means that it is necessary to define a fixed frame in world coordinates, $\mathcal{R} = (\mathbf{o}_g, \mathbf{x}, \mathbf{y}, \mathbf{z})$, referred to as the global frame. Here $\mathbf{o}_g$ is a fixed point, $\mathbf{z}$ points upwards, in the opposite direction of the gravity force, and $\mathbf{x}$ and $\mathbf{y}$ are fixed vectors in predefined directions, perpendicular to each other and $\mathbf{z}$.

In addition to the seven generalized coordinates needed to represent the configuration of the arm, the position and pose of the base is represented by three variables, as shown in Figure 2.6, giving the slave manipulator a total of 10 generalized coordinates. By using Denavit-Hartenberg convention, 10 frames are defined, in addition to the global and a zero frame.

The origin of frame 0, $\mathbf{o}_0$, is located at the global origin, $\mathbf{o}_g$, while the axis and the rest of the frames are found using the algorithm in Section B.1. The DH-parameters for the mobile manipulator can be found in Table 3.1, where q_1 and q_2 are considered as two prismatic joints.

Since the mobile manipulator consists of 10 generalized coordinates, the

Link	a_i	α_i	d_i	θ_i
0	0	$\frac{\pi}{2}$	0	$\frac{\pi}{2}$
1	0	$\frac{\pi}{2}$	q_1	$\frac{\pi}{2}$
2	0	$\frac{\pi}{2}$	q_2	$\frac{\pi}{2}$
3	$-a_3$	π	0	$q_3 + \pi$
4	0	$\frac{\pi}{2}$	$-d_4$	q_4
5	0	$-\frac{\pi}{2}$	0	q_5
6	0	$\frac{\pi}{2}$	$-d_6$	q_6
7	0	$-\frac{\pi}{2}$	0	q_7
8	0	$\frac{\pi}{2}$	$-d_8$	q_8
9	0	$-\frac{\pi}{2}$	0	q_9
10	0	$\frac{\pi}{2}$	$-d_{10}$	q_{10}

Table 3.1: Link Parameters for Mobile Manipulator

transformation of the end-effector is given by

$$T_g^n = T_g^{10} = A_0 A_1 ... A_{10}, \tag{3.43}$$

where T_g^n refers to the transformation from frame n to the global frame and the homogeneous transformation matrices A_i are found by inserting the respective parameters in expression for the matrix B.1. The resulting matrices can be found in Section B.4, while the zero configuration, with the given transformations, can be seen in Figure 3.7.

3.5.2 Mobile Base

Because of the wheeled base, the slave manipulator is introduced to nonholonomic constraints[1]. The configuration of the mobile base can be defined with generalized coordinates $\mathbf{q}_b$, equal to the location, which can be defined with three operational coordinates

$$\xi_b = [\xi_{b1} \quad \xi_{b2} \quad \xi_{b3}]^T = [x \quad y \quad \phi]^T = \mathbf{q}_b, \tag{3.44}$$

[1]Described in Section 2.3.2

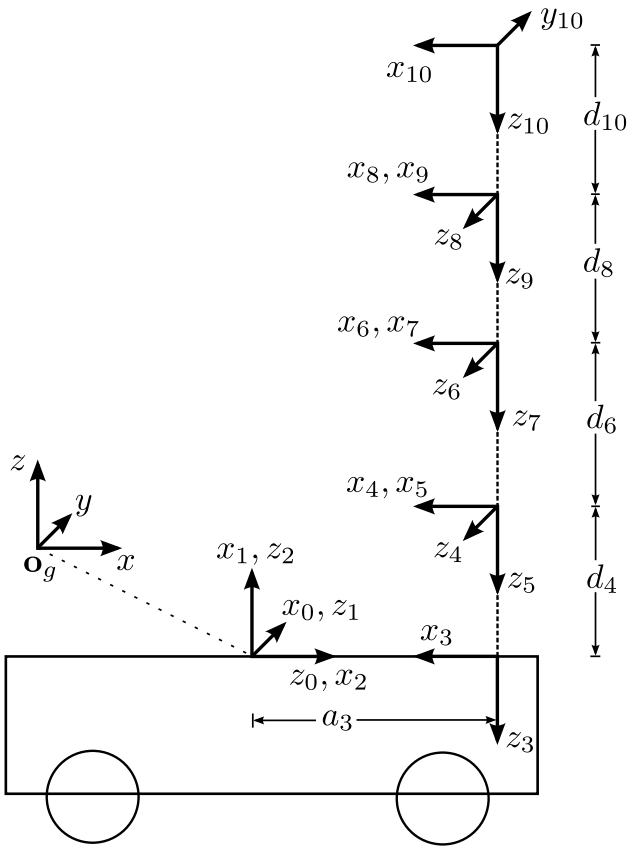


Figure 3.7: The zero configuration of the mobile manipulator.

where x and y describe the position relative to $\mathbf{o}_g$, and ϕ the angle between the direction of the base and the x-axis, defined in $\mathcal{R}$.

The control inputs for the base are

$$\mathbf{u}_b = [u_{b1} \quad u_{b2}]^T = [v \quad \omega]^T, \tag{3.45}$$

where v and ω are the linear and angular velocities, respectively. The velocities are related to the generalized coordinates by the configuration differential kinematic model given as $\dot{\mathbf{q}}_b = \mathbf{S}(\mathbf{q}_b)\mathbf{u}_b$, where

$$\mathbf{S}(\mathbf{q}_b) = \begin{bmatrix} \cos\phi & 0 \\ \sin\phi & 0 \\ 0 & 1 \end{bmatrix}. \tag{3.46}$$

As mentioned in Section 2.3.2, the nonholonomic constraints can be described by

$$\dot{x}\sin\phi - \dot{y}\cos\phi = 0. \tag{3.47}$$

In matrix form, the constraints can be written as

$$\mathbf{G}_b(\mathbf{q}_b)\dot{\mathbf{q}}_b = 0, \tag{3.48}$$

where $\mathbf{G}_b(\mathbf{q}_b) = [\sin\phi \quad -\cos\phi \quad 0]$.

3.5.3 Robot Arm

For the robot arm, the configuration is represented by 7 generalized coordinates,

$$\mathbf{q}_a = [q_{a1} \quad q_{a2} \quad \dots \quad q_{a7}]^T, \tag{3.49}$$

and the location of the end-effector, relative to the mobile base, is represented by the operational coordinates $\xi_a = \mathbf{f}_a(\mathbf{q}_a)$.

The forward differential kinematics are given as

$$\dot{\xi}_a = \mathbf{J}_a(\mathbf{q}_a)\dot{\mathbf{q}}_a, \tag{3.50}$$

where the Jacobian $\mathbf{J}_a$ is given as

$$\mathbf{J}_a(\mathbf{q}_a) = \frac{\partial \mathbf{f}_a(\mathbf{q}_a)}{\partial \mathbf{q}_a}. \tag{3.51}$$

The kinematics for the mobile base are combined with kinematics for the robot arm, the result will be the next topic.

3.5.4 Mobile Manipulator

When mounting the manipulator on the mobile base, the configuration is defined by 3+7 generalized coordinates,

$$\mathbf{q} = [q_1 \quad q_2 \quad ... \quad q_{10}]^T = [x \quad y \quad \phi \quad q_{a1} \quad q_{a2} \quad ... \quad q_{a7}]^T. \tag{3.52}$$

The matrix $\mathbf{G}(\mathbf{q})$ can be defined, by including the constraints (3.48), as

$$\mathbf{G}(\mathbf{q}) = [\mathbf{G}_b(\mathbf{q}_b) \quad 0 \quad ... \quad 0], \tag{3.53}$$

making it possible to write

$$\mathbf{G}(\mathbf{q})\dot{\mathbf{q}} = 0. \tag{3.54}$$

The end-effector position and orientation, relative to $\mathcal{R}$, are characterized by six operational coordinates $\xi = [\xi_1 \quad \xi_2 \quad ... \quad \xi_6]^T$. By combining the expression for the nonholonomic constraints (3.53) and for the forward differential kinematics (3.6), the velocity constraints can be written as

$$\begin{bmatrix} \mathbf{G}(\mathbf{q}) \\ \mathbf{J}(\mathbf{q}) \end{bmatrix} \dot{\mathbf{q}} = \begin{bmatrix} 0 \\ \dot{\xi} \end{bmatrix}. \tag{3.55}$$

By defining the vector of the velocities for the system as $\mathbf{u} = [\mathbf{u}_b^T \ \dot{\mathbf{q}}_a^T]^T$, the configuration differential kinematic model for the mobile manipulator can be written as $\dot{\mathbf{q}} = \mathbf{M}(\mathbf{q})\mathbf{u}$, with

$$\mathbf{M}(\mathbf{q}) = \begin{bmatrix} \mathbf{S}(\mathbf{q}_b) & 0 \\ 0 & \mathbf{I}_7 \end{bmatrix}, \tag{3.56}$$

where $\mathbf{S}$ is defined in (3.46). By taking into account that $\mathbf{G}(\mathbf{q})\mathbf{M}(\mathbf{q}) = 0$, it is possible to write

$$\dot{\xi} = \bar{\mathbf{J}}(\mathbf{q})\mathbf{u}, \tag{3.57}$$

where $\bar{\mathbf{J}}(\mathbf{q}) = \mathbf{J}(\mathbf{q})\mathbf{M}(\mathbf{q})$.

3.5.5 Forward Differential Kinematics

The mobile manipulator will operate in two modes, one where the base moves and one where the base is fixed. This makes it necessary to find both the Jacobian for the whole system, $\mathbf{J}$, and for the arm alone, $\mathbf{J}_a$. Unlike the Jacobian described in Section 3.5.3, the new Jacobian for the arm will also depend on the configuration of the base, such that $\mathbf{J}_a = \mathbf{J}_a(\mathbf{q})$.

The Jacobian, $\mathbf{J}$, is found with the procedure described in Section 3.2, while $\mathbf{J}_a$ is found by removing the last two columns of $\mathbf{J}$ (which corresponds to imposing $v = \omega = 0$).

The inverse differential kinematics are more a choice of design and are covered by the next chapter, after a short summary of this chapter.

3.6 Summary

The kinematics for the master and slave robot, presented in this chapter, is used to transform the movement between the operational and joint-space. The use of operational space is necessary because the master and slave manipulator is kinematically different. This is a result of both different shapes and joint numbers.

The forward kinematics describe the position and orientation in Cartesian coordinates, based on the joint configuration. The forward differential kinematics transform the velocity from joint-space to operational space, while inverse differential kinematics are used to find the desired joint velocity, based on reference velocity in operational space.

Due to the nonholonomic constraints, a new form of Jacobian is used. This complicates the calculation and transformation between the different coordination systems.

Since the slave robot is redundant, it is possible to impose several tasks simultaneously. The design of the velocity vector $\mathbf{u}^*$ and the priority of

different tasks, are part of the control architecture discussed in the next chapter.

Chapter 4

Control Architecture

This chapter describes the structure of the controllers used in this study. In addition to a controller for the mobile manipulator, it is also necessary to design a controller for the haptic joystick, calculating a proper force feedback to the human operator.

Several aspects are considered when designing the controllers, where obtaining stability often is considered as the main objective. When the system is stable, the next step is to ensure output synchronization, that is, making the manipulator achieve a desired joint angle. There has been much research on stability with time delay and tracking for kinematically equal systems, and the theory this control architecture is based on is summarized in Section 2.5.2.

For the kinematically different slave and master manipulator, the desired joint angles are often based on a task specified in operational space. To get from operational space to joint space, it is necessary to use inverse kinematics for angles and positions, and inverse differential kinematics for velocities. Different solutions to find the transformation from velocities in operational space to velocities in joint space can be found in Section 3.3.

The theory this control architecture is based on, to achieve both stability and output synchronization, can be found in Section 2.5. The rest of this

chapter is divided into three sections, the first describing the slave controller, used for the mobile manipulator, the second describing the master controller, used for the human operated joystick, and the last ensuring overall stability of the system.

4.1 Slave Controller

The slave controller is partitioned into three parts. The first step is to convert a reference velocity for the end-effector, specified in operational space, to a desired velocity in joint space, and will be the topic of 4.1.1. Since the mobile manipulator is redundant, it is possible to optimize an additional function to get a desired pose. This will be discussed in Section 4.1.2. The last step is to calculate a desired force, based on a desired velocity.

In this study, the part that calculates the generalized force, τ_s, based on a desired control input, $\mathbf{u}^*$, is already provided. This part is some sort of PID-controller, but more detailed information is restricted. The control input is used instead of the generalized velocity, due to the nonholonomic constraints.

4.1.1 Inverse Differential Kinematic

The desired velocity in joint space is calculated from a set of velocity tasks in operational space, $\dot{\mathbf{t}}_i$. In this study, only the joint limits, end-effector position and manipulability are considered when calculating $\mathbf{u}^*$. Even though the joint limits are not specified in operational space, it is possible to design a transformation matrix $\bar{\mathbf{J}}_l$ satisfying $\dot{\mathbf{t}}_l = \bar{\mathbf{J}}_l(\mathbf{q}_s)\mathbf{u}$ and given by

$$\bar{\mathbf{J}}_l(\mathbf{q}_s) = \begin{bmatrix} d_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & d_2 & 0 & \cdots & 0 & 0 \\ 0 & \ddots & \ddots & 0 & \vdots & \vdots \\ 0 & \cdots & 0 & d_7 & 0 & 0 \end{bmatrix}. \tag{4.1}$$

Here $\mathbf{q}_s$ is the generalized coordinate describing the mobile manipulator and the entries d_i are found as

$$d_i = \begin{cases} 1 & \text{if } (q_{s,i} > h_i - \epsilon) \ \cap (\dot{q}_{s,i} > 0) \\ -1 & \text{if } (q_{s,i} < l_i + \epsilon) \ \cap (\dot{q}_{s,i} < 0) \\ 0 & \text{otherwise} \end{cases}, \tag{4.2}$$

with $\epsilon > 0$, h_i and l_i as the upper and lower limit for joint i, respectively, and the elements of the velocity task defined as

$$\dot{t}_{l,i} = \delta < 0. \tag{4.3}$$

The tasks are organized in a hierarchy, where the joint limits have the highest priority and manipulability the lowest. By using the pseudoinverse of $\bar{\mathbf{J}}_i$ for each task, the control input can be written as

$$\mathbf{u}^* = \bar{\mathbf{J}}_l^\dagger(\mathbf{q}_s)\dot{\mathbf{t}}_l + \mathbf{N}_l(\bar{\mathbf{J}}_e^\dagger(\mathbf{q}_s)\dot{\xi}^* + \mathbf{N}_e\mathbf{w}), \tag{4.4}$$

where $\bar{\mathbf{J}}_l$ is given by (4.1) and $\dot{\mathbf{t}}_l$ by (4.3), and the null space of the joint limits defined as $\mathbf{N}_l = \mathbf{I}_n - \bar{\mathbf{J}}_l^\dagger(\mathbf{q}_s)\bar{\mathbf{J}}_l(\mathbf{q}_s)$. Here the pseudoinverse is found using singular value decomposition (SVD). $\dot{\xi}^*$ is the desired end-effector velocity, and is related to the control input through the matrix $\bar{\mathbf{J}}_e$, derived in Section 3.5. $\mathbf{N}_e$ spans the null space of the transformation matrix for the desired end-effector velocity, and is given as $\mathbf{N}_e = \mathbf{I}_n - \bar{\mathbf{J}}_e^\dagger(\mathbf{q}_s)\bar{\mathbf{J}}_e(\mathbf{q}_s)$.

The desired end-effector velocity, $\dot{\xi}^*$, is given relative the global frame, $\mathcal{R}$, and found as

$$\dot{\xi}^* = R_g^e \dot{\xi}_e^*, \tag{4.5}$$

where $\dot{\xi}_e^*$ is the desired end-effector velocity given relatively to the end-effector frame $\mathcal{R}_n$, and R_g^e is the rotational matrix found as the 3×3 matrix in the upper left corner of T_g^n. The expression for T_g^n can be found in Section 3.5.1, while $\dot{\xi}_e^*$ is calculated by the master controller and discussed in further detail in Section 4.2. The orientation of the end-effector frame can be seen in

Figure 3.7, where z_n is defined as the axis from the end-effector towards the previous joint.

The last term in the expression for the control input (4.4), $\mathbf{w}$, is the gradient of a function representing the pose of the slave robot. The calculation of this gradient is the next topic.

4.1.2 Optimization Criteria

The task with lowest priority is to maintain a good posture. Different criteria can be used to define a good posture, such as manipulability, distance to joint limits or equal joint angles.

In this study the manipulability is one of the optimization criteria. Since the slave robot can operate both with the base moving and fixed, the manipulability is optimized for both the entire mobile manipulator and the robot arm itself. The manipulability measure can be written as

$$\mathcal{P}(\mathbf{q}_s) = \alpha(\mathbf{q}_s)\mathcal{P}_{b+a}(\mathbf{q}_s) + (1 - \alpha(\mathbf{q}_s))\mathcal{P}_a(\mathbf{q}_{s,a}), \tag{4.6}$$

with

$$\mathcal{P}_{b+a}(\mathbf{q}_s) = \sqrt{1 - \frac{\sigma_m^2}{\sigma_1^2}} \tag{4.7}$$

$$\mathcal{P}_a(\mathbf{q}_{s,a}) = \frac{|q_{s,4}|}{2\pi}\sigma_{a1}\sigma_{a2}...\sigma_{am}. \tag{4.8}$$

Here $\mathcal{P}_{b+a}$ and $\mathcal{P}_a$ are based on w_5 and w in Section 2.2, σ_i are the singular values of $\bar{\mathbf{J}}_e$, σ_{ai} the singular values of $\bar{\mathbf{J}}_a$ and the scalar function, $\alpha(\mathbf{q}_s)$, is chosen as

$$\alpha = \begin{cases} c_\alpha & \text{if base is moving} \\ 0 & \text{if base is fixed,} \end{cases} \tag{4.9}$$

where $c_\alpha > 0$. To optimize the manipulability, it is necessary to find the

gradient $\mathbf{w}_1$ given as

$$\mathbf{w}_1 = \kappa_1 \left(\frac{\partial \mathcal{P}(\mathbf{q}_s)}{\partial \mathbf{q}_s} \mathbf{M} \right)^T, \tag{4.10}$$

where $\kappa_1 > 0$ is a weighting variable, and $\mathbf{M}$ is defined in Section 3.5 and satisfies $\dot{\mathbf{q}}_s = \mathbf{M}(\mathbf{q})\mathbf{u}$.

In addition to high manipulability, it is desirable to make $q_{s,6}$ and $q_{s,8}$ move towards 0. This corresponds to favor an "elbow up"-pose for the arms fourth joint. The gradient for the second optimization criteria, $\mathbf{w}_2$, can be written as

$$\mathbf{w}_2 = -\kappa_2 [0,\ 0,\ 0,\ 0,\ q_{s,6},\ 0,\ q_{s,8},\ 0,\ 0]^T, \tag{4.11}$$

where κ_2 is a weighting variable.

The optimization criteria, $\mathbf{w}$, is given as the sum of $\mathbf{w}_1$ and $\mathbf{w}_2$. By inserting $\mathbf{w}$ into (4.4), the new expression for the control input is given as

$$\mathbf{u}^* = \bar{\mathbf{J}}_l^\dagger(\mathbf{q}_s)\dot{\mathbf{t}}_l + \mathbf{N}_l(\bar{\mathbf{J}}_e^\dagger(\mathbf{q}_s)\dot{\xi}^* + \mathbf{N}_e(\mathbf{w}_1 + \mathbf{w}_2)), \tag{4.12}$$

which is used as input for the provided force controller.

The generalized coordinates $\mathbf{q}_s$ are measured and sent to the master controller, which is described in the next section.

4.2 Master Controller

The master controller calculates force feedback provided to the human operator through the haptic joystick, in addition to generate a desired velocity, in operational space, for the end-effector of the mobile manipulator.

First in this section, the desired linear and angular velocities are derived, followed by the calculation of the force feedback. In the rest of this section, the term *end-effector* refers to the last link of the mobile manipulator.

4.2.1 Calculation of Desired Velocity

The desired end-effector velocity relative the end-effector frame, $\dot{\xi}_e^*$, is partitioned into a desired linear and angular velocity, $\mathbf{v}_e^*$ and ω_e^*, respectively. Each of the velocities are found as the error between a desired change and the actual change in location, times a proportional gain, resulting in

$$\mathbf{v}_e^* = -k_3 \left(\tilde{\mathbf{p}}_e(t, t_0) - k_1 \tilde{\mathbf{p}}_e^*(t, t_0) \right) \tag{4.13}$$

$$\omega_e^* = -k_4 \left(\tilde{\theta}_e(t, t_0) - k_2 \tilde{\theta}_e^*(t, t_0) \right). \tag{4.14}$$

Here $\tilde{\mathbf{p}}_e(t, t_0) = R_e^g(\mathbf{p}_g(t - T(t)) - \mathbf{p}_g(t_0 - T(t_0)))$ is the difference in end-effector position given in local coordinates, and $\tilde{\theta}_e$ is the Euler angle between the end-effector frame measured at time $t - T(t)$ and at time $t_0 - T(t_0)$. Here $R_e^g = \left(R_g^e\right)^T$ is the rotation from the end-effector frame, $\mathcal{R}_n$, to the global frame, $\mathcal{R}$. $\mathbf{p}_g$ is the global position of the end-effector, t is the time, $T(t)$ is the time-varying delay, and $k_i \in \mathbb{R}_{>0}$.

The desired change in end-effector position and orientation are based on a change in the joystick position and rotation, where the change in joystick position, relative the joystick frame, is given as

$$\tilde{\mathbf{p}}_e^* = \bar{R}_e^g \tilde{\mathbf{p}}_g^*(t, t_0), \tag{4.15}$$

with

$$\tilde{\mathbf{p}}_g^*(t, t_0) = \bar{\mathbf{p}}_g(t) - \bar{\mathbf{p}}_g(t_0), \tag{4.16}$$

and the change in rotation, $\tilde{\theta}_n^*$, is given as the Euler angle between the joystick orientation at time t and at time t_0. The rotation matrix $\bar{R}_g^e = \left(\bar{R}_e^g\right)^T$ describes the rotation of the joystick relative to the global joystick frame, and $\bar{\mathbf{p}}_g$ is the joystick position relative the global joystick frame.

To enhance the precision of the movement from the human operator, the change in joystick position and rotation are scaled by k_1 and k_2, respectively. That is, a large movement with the joystick corresponds to a smaller movement of the mobile manipulator, relative to their workspace. The initial

location of the end-effector and joystick, at time t_0, is measured when the blue button on the Phantom Omni, seen in Figure 3.4, is pressed. As long as the button is pressed, the desired change of location is the scaled difference between the measured and initial joystick location. When the button is released, the desired change is set to zero, and when pressed again, a new initial location is set.

4.2.2 Force Feedback

The force feedback is used to provide the human operator with additional details about the state of the mobile manipulator. These details may include information about the distance to the surrounding obstacles, the force applied by the end-effector at an object, and the deviation between the desired and actual location and velocity.

In this study, the force feedback reflects the distance between the desired and measured end-effector position. The reason for not including orientation in the feedback, is that the Phantom Omni only provides force in three dimensions, specified in Cartesian coordinates. The force is found as the position error multiplied by a proportional constant, K_m.

In addition to the proportional part, a damping of the joystick velocity is introduced. The expression for the force generated by the master manipulator, is given as

$$\mathbf{F}_m = -K_m \epsilon_m - B_m \dot{\tilde{\mathbf{p}}}_g, \tag{4.17}$$

with

$$\epsilon_m = \bar{R}_g^e \left(\tilde{\mathbf{p}}_e^*(t, t_0) - \frac{1}{k_1} \tilde{\mathbf{p}}_e(t, t_0) \right). \tag{4.18}$$

Here ϵ_m corresponds to an error given in the global joystick frame.

The values of k_i, K_m and B_m affect the stability of the system, and is discussed in the next section.

4.3 Ensuring System Stability

As long as the control architecture for the part that converts the desired control input to torque is unknown, any analytical stability analysis is difficult. Since the derivative term in a PID-controller only makes a system more stable [123] and because a PI-regulator is more similar to the controllers presented in Section 2.5, the provided control architecture is assumed to be dominated by a PI-term. The alleged PI-controller is on the form

$$\tau_s = -K_p(\mathbf{u}(t) - \mathbf{u}^*(t)) - K_i \int_0^t (\mathbf{u}(\sigma) - \mathbf{u}^*(\sigma))\,\mathrm{d}\sigma, \tag{4.19}$$

while the control input $\mathbf{u}^*$, which is calculated based on a the deviation in position and orientation between the end-effector and joystick, is assumed to be radially unbounded and satisfying

$$\|\mathbf{u}^*\| \leq \delta \quad \text{if} \quad \|\epsilon_s\| = 0,$$

where $0 \leq \delta \in \mathbb{R}$ is a small number and ϵ_s is the position and orientation error. A function V is said to be radially unbounded if the function satisfy the following condition [124]:

$$V(x) \to \infty \quad \text{as} \quad \|x\| \to \infty.$$

The assumptions concerning $\mathbf{u}^*$ implies that the control input gets large if the error gets large and are bounded when the error approaches zero, which means that the system is asymptotically stable if the error is zero. This is only true if the constants κ_1 in (4.10) and κ_2 in (4.11) are small enough.

With the proposed assumption, the slave controller can be considered as a P + d controller on the form

$$\tau_s = -K_s\epsilon_s - B_s\mathbf{u}(t). \tag{4.20}$$

Here ϵ_s depends on the location error and $B_s = K_p$.

With the assumption that the slave controller being a P + d controller, the system is stable if satisfying [99]

$$\lambda_1 B_m B_s > ({}^*T_m^2 +^* T_s^2) K_m K_s, \tag{4.21}$$

where ${}^*T_i$ are the upper bound on the variable time-delay, B_i and K_i defined as in (4.17) and (4.20), and λ_1 is a constant. Since B_s is given, the variable K_s must be chosen to ensure stability of the mobile manipulator, regardless of the force feedback. When K_s is chosen, the relationship between B_m and K_m is given as

$$\lambda_2 B_m > K_m,$$

where λ_2 is a constant depending on λ_1, B_s, K_s and ${}^*T_i$.

The controllers proposed in this chapter are implemented on a personal computer, and tested in a simulated environment with a Phantom Omni. The next chapter covers the main elements of the framework needed to test the controllers.

Chapter 5

System Overview

To test the control architecture presented in Chapter 4, it is necessary to implement the architecture and design a framework to emulate a bilateral teleoperation system. This chapter gives the reader an overview of the developed program and the framework used to analyze the design and choices made.

The framework consists of a master manipulator, covered in Section 5.1, an application programming interface (API), described in Section 5.2, and a physics-engine-based simulator, presented in Section 5.5.

The API, control scheme and communication with the simulator are implemented using Robot Operating System (ROS), where a short description of ROS is found in Section 2.6. The part of the implementation developed in this study, calculates the desired slave velocity and force feedback. Included in the developed program are the control architecture, proposed in Chapter 4, the kinematics of the manipulators, derived in Chapter 3, and communication with the joystick. The part of the program developed in this study is divided in two, called *Master Controller* and *Slave Controller*. Section 5.3 presents the master controller part, while the part called slave controller is described in Section 5.4. Section 5.6 summarizes the most important elements from this chapter.

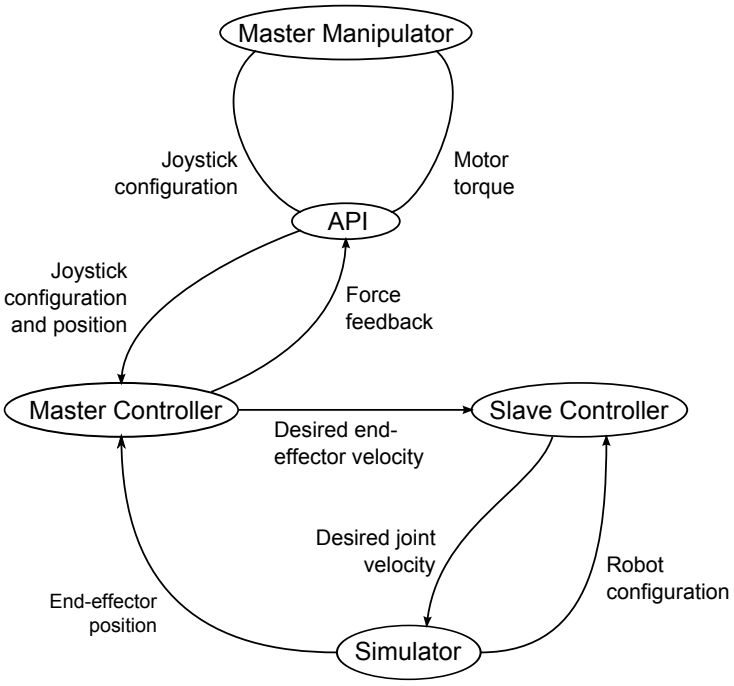


Figure 5.1: Overview of the framework and implemented program. *Master Manipulator* corresponds to a human operated haptic joystick, while *Simulator* is a physics-engine-based simulator.

An overview of the system is shown in Figure 5.1, while a more detailed description of the implementation can be found in Chapter 6.

5.1 Master Manipulator

The master manipulator in this study is a haptic joystick, which can provide the operator with force feedback in 3 degrees of freedom (DOF), and measure the applied movement in 6 DOF. The joystick used is a Phantom Omni, shown in Figure 2.8, and is made by Sensable.

The kinematics for the joystick are derived in Section 3.4, and are used to find the position and orientation of the pen, shown in Figure 3.4, based on the measurements of the joint angles. The measurements are sent to a personal computer (PC), which is used to analyze the values and calculate a force

feedback.

To communicate between the joystick and the PC, an application programming interface (API) is used. The next section includes a brief description of the API used in this study.

5.2 Application Programming Interface

For the program designer to easier understand the messages from the joystick and to assign feedback, an application programming interface (API) is used. For the Phantom Omni, OpenHaptics toolkit is provided from Sensable, which includes Haptic Device API (HDAPI) and Phantom Device Drivers (PDD).

HDAPI is a low-level API, and communicates with the Phantom Omni through the PDD. In HDAPI, the forward and inverse kinematics regarding position of the pen are precalculated, allowing the programmer to work directly with position and force vectors in Cartesian coordinates. For the orientation of the pen, no such kinematic calculation is included, forcing the developer to calculate the total rotation based on the measurements of the joint angles from the joystick.

The kinematics for the haptic joystick, as well as the communication with the joystick are implemented using ROS. The main structure of this program is the next topic.

5.3 Master Controller

The part of the program called master controller, calculates the desired end-effector velocity for the mobile manipulator and the force feedback, sent to the human operator. The inputs to this part are the position and configuration of the joystick, extracted from the API, and the position of the end-effector, generated by the simulator. The calculations are based on the

kinematics of the joystick, found in Section 3.4, and the controller for the joystick, designed in Section 4.2.

The force feedback is sent back to the human operator by communicating through the API, while the desired end-effector velocity is sent to the slave controller, presented next.

5.4 Slave Controller

The other part of the developed program, called slave controller, calculates the desired joint velocities for the mobile manipulator. The desired joint velocities are calculated by using the controller for the mobile manipulator, presented in Section 4.1, and the kinematics of the slave robot, derived in Section 3.5. The inputs to this part are the desired end-effector velocity, generated by the master controller, and the configuration of the slave robot.

The slave controller sends the desired joint velocities to a physics-engine-based simulator, which is described next.

5.5 Physics-Engine-Based Simulator

To generate a response from the desired joint velocity, a physics-engine-based simulator is used. The simulator calculates an end-effector position and joint angles for the slave robot, based on the desired control input and a model of the mobile manipulator. The movements of the slave robot are represented graphically in a virtual world.

5.6 Summary

The elements presented in this chapter form the frame used to implement the proposed control architecture. The framework includes a Phantom Omni, which communicates to a personal computer using an API, and a physics-engine-based simulator. The API and the control architecture are implemented using ROS, with the key elements from the implementation of the developed program described in the next chapter.

Chapter 6

Implementation

This chapter describes the main parts of the implemented program, which include the control architecture designed in Chapter 4, the communication to the joystick, and the simulator for the mobile manipulator. The program is implemented using ROS, where a short description of ROS and the most important terms are found in Section 2.6.

While the program is implemented in ROS, the code is written in Python and C++ programming language, with the use of libraries and features included in ROS. As described in Chapter 5, a ROS-based program is designed to communicate through the API to the joystick, and to a physics-engine based simulator. In addition, the program calculates the desired force feedback and joint velocities for the mobile manipulator.

The developed program is based on a framework provided by SINTEF, which includes forwarding of the joystick position from and force feedback to the API, and calculation of the linear velocity of the joystick. Included in the framework is also the physics-engine based simulator, the ability to send a desired joint velocity for the robot arm to the simulator, and the model of the mobile manipulator, used in the simulation.

In this study, the provided framework is altered to forwarding the joystick configuration from the API, and to send desired velocities for both the robot

arm and mobile base to the simulator. The developed program consists of two parts, called master and slave controller, both communicating with the simulator.

The implemented program consists of several nodes, as explained in Section 2.6, where each node has a specific task. The structure of the program is shown in Figure 6.1, where the encircled names correspond to nodes, and the arrows between the nodes correspond to messages.

First, in Section 6.1, the simulator, described in Section 5.5, is presented in more detail. Section 6.2 describes the implementation of the master controller from Section 5.3, while Section 6.3 presents the implemented nodes, corresponding to the slave controller in Section 5.4

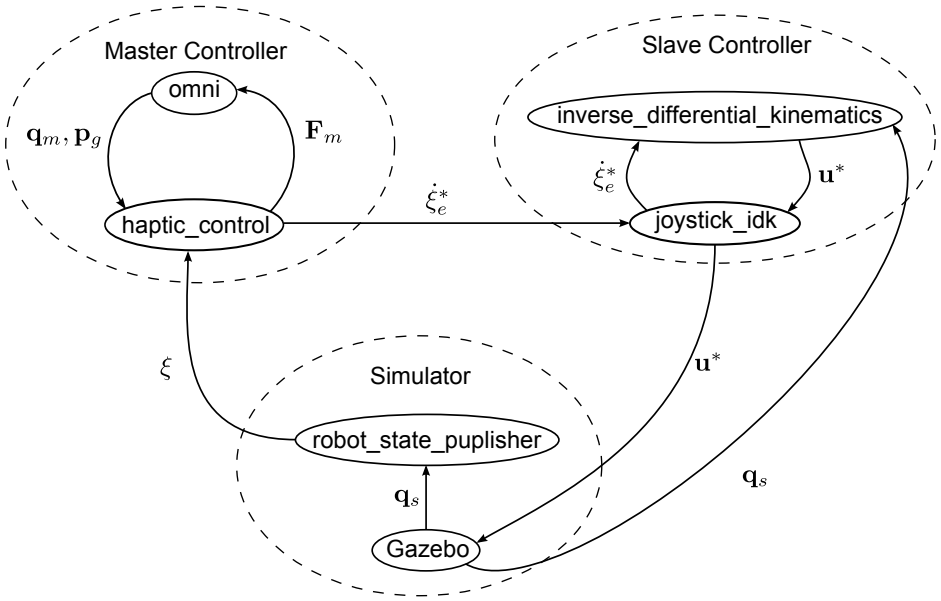


Figure 6.1: The nodes (encircled) and messages (arrows) in the implemented program.

6.1 Physics-Engine-Based Simulator

This section describes the conversion from a desired control input to the configuration and end-effector position of the virtual mobile manipulator. For the robot arm the control input corresponds to desired joint velocities, and desired linear and rotational velocity for the mobile base.

The simulator part of the program consists of two nodes. The first node, called Gazebo, is the actual simulator, and calculates the joint angle for a virtual mobile manipulator based on a model of the robot to be controlled. The second node, robot_state_publisher, calculates the end-effector position for the mobile manipulator.

6.1.1 Gazebo-node

To graphically represent and simulate the movement of the mobile manipulator, a third party physics-engine-based simulator, called Gazebo, is used. Gazebo is a multi-robot simulator with dynamics, which presents robots and their movements in a three-dimensional virtual world, as well as generating realistic sensor feedback [125]. The simulator gives the human operator a visual feedback, in addition to the force feedback from the joystick.

The total movement consists of a set of local transformations, where each transformation is given as a translation and rotation of a frame, relative the parent frame. The frames, representing the position and orientation of the different parts of the virtual mobile manipulator, are shown in Figure 6.2, where the arrows points from a parent frame to its children frames. Here /map and /odom are fixed to the environment, /base_footprint and /base_link to the mobile base, /arm_i_link to the robot arm, and the rest to each of the wheels.

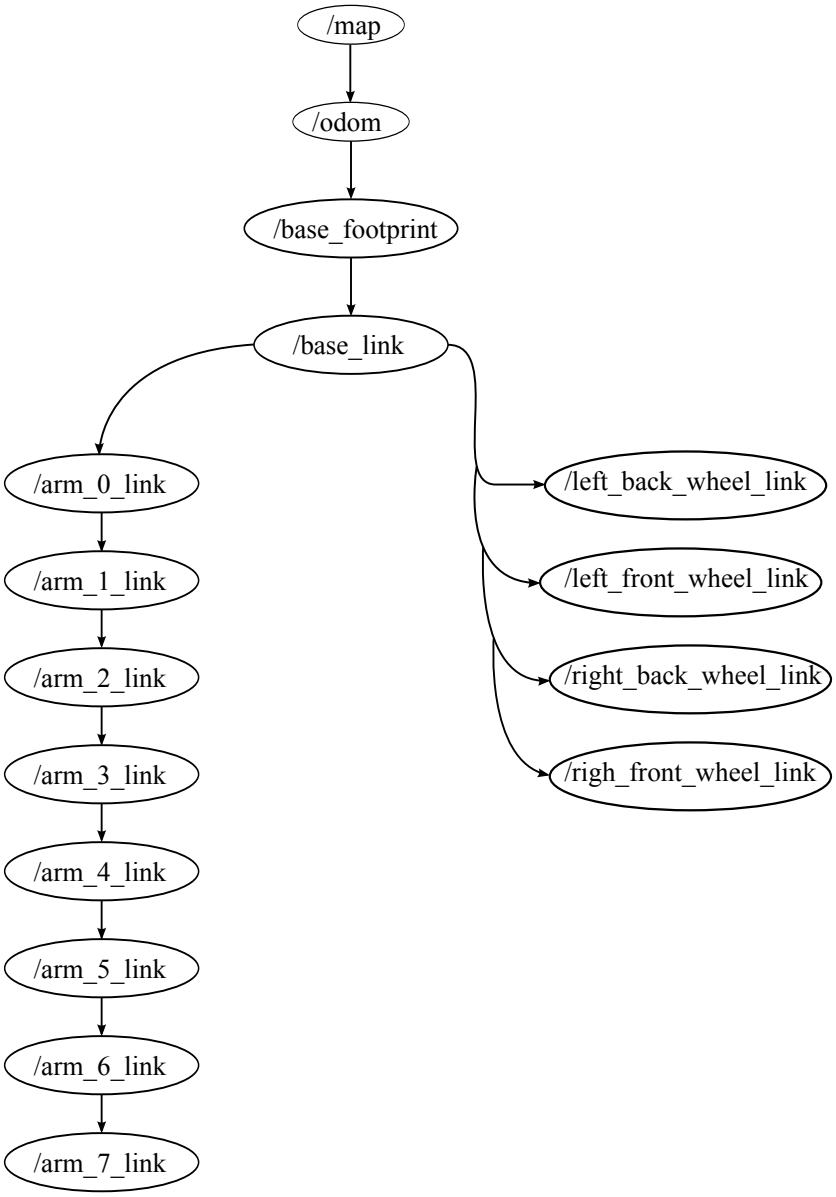


Figure 6.2: The frames representing the configuration of the virtual mobile manipulator. */map* and */odom* are fixed to the environment, while the rest of the frames are fixed to a part of the slave robot.

The simulator publishes the configuration of the virtual mobile manipulator on a topic, where it is read by, among others, the robot_state_publisher-node.

6.1.2 robot_state_publisher-node

The robot_state_publisher-node is included in ROS, and publishes the configuration of the virtual mobile manipulator to tf, see Section 2.6, making the position and orientation of the frames available to all components of the program that use tf [126]. One of these frames is fixed to the slave end-effector, making it possible to extract the position and orientation of the end-effector from tf.

The end-effector position and orientation are used by the master controller to generate a desired end-effector velocity, further discussed in the next section.

6.2 Master Controller

This section describes the part of the program that corresponds to the master controller, designed in Section 4.2, and consists of the following nodes: *omni* and *haptic_control*. The main purpose of this part is to calculate a desired end-effector velocity, based on the end-effector position and orientation, and the interaction with a human operator.

To interact with the human operator, it is necessary to communicate with the Phantom Omni. That is, measuring the position and orientation of the haptic device, as well as sending a desired force back to the device. This communication goes through the omni-node, presented next.

6.2.1 omni-node

Prior to this study, the omni-node is designed to forward the desired force feedback to and measure the joystick position from an API, in addition to calculate an average linear joystick velocity. The omni-node uses HDAPI, described in Section 5.2, to acquire the position of the joystick pen in Cartesian coordinates, as well as forwarding a desired force feedback from the haptic_control-node to the joystick. The position is found by HDAPI by using a precalculated forward kinematic model of the joystick.

In addition to the position, it is in this study desirable to find the orientation and linear velocity of the pen. Neither the velocity nor the orientation is included in HDAPI, but for the orientation it is possible to extract the generalized coordinates describing the configuration of the joystick, $\mathbf{q}_m$. The angles are partitioned into J_1, J_2 and J_3, called joint angles, and J_4, J_5 and J_6, called gimbal angles, where J_i are seen in Figure 3.1 and Figure 3.4. The angles can be extracted from HDAPI and copied to `var_1` and `var_2`, using the following command in C++

```
hdGetDoublev(HD_CURRENT_GIMBAL_ANGLES, var_1);
hdGetDoublev(HD_CURRENT_JOINT_ANGLES, var_2);
```

while the position is copied to `var_3` using

```
hdGetDoublev(HD_CURRENT_POSITION, var_3);
```

where `var_i` are local variables. A function for calculating the average velocity of the joystick is already provided in the node, but not published and made available for other nodes. This is solved by extending the message between the omni- and haptic_control-node to include velocity.

A desired force feedback to the human operator, `force`, is sent to the joystick by writing

```
hdSetDoublev(HD_CURRENT_FORCE, force);
```

where `force` is a local variable.

This node communicates with the rest of the program through haptic_control, by publishing the configuration of the joystick, as well as the position and linear velocity, and subscribing to the topic with the desired force feedback.

6.2.2 haptic_control

The haptic_control-node is designed to calculate a desired end-effector velocity, relative to the end-effector frame, and force feedback to the human operator. The calculations are based on the position and orientation of the end-effector and joystick, as well as the linear velocity of the joystick. Both the joystick position and linear velocity are extracted from the omni-node, while the joystick orientation is found from the derived kinematics of the master manipulator, described in Section 3.4, and the joystick configuration from the omni-node.

Force Feedback

The force feedback is linearly dependent on the position error and the linear velocity of the human operated joystick, given by equation (4.17). The starting point for the joystick position, $\bar{\mathbf{p}}_g(t_0 - T(t_0))$, and orientation, $\bar{\theta}_g(t_0 - T(t_0))$, are set as the corresponding measurements at the time when the blue button on the Phantom Omni is pressed, and kept the same until the button is released again. The linear velocity of the joystick, $\dot{\mathbf{p}}_g$, is found from the message received from the omni-node.

The calculated force is compared with a predefined maximum force, before it is published and made available to the rest of the program, including the omni-node. In addition to finding the force feedback, sent to the human operator, the haptic_control-node calculates the desired end-effector velocity.

Desired End-Effector Velocity

The desired end-effector velocity, $\dot{\xi}_e^*$, is found as the combination of the desired linear velocity (4.13) and desired angular velocity (4.14), where k_1 and k_2 are scaling factors, and k_3 and k_4 are proportional gains, depending on the time delay. The end-effector position, $\mathbf{p}_g(t - T(t))$, and orientation, $\theta_g(t - T(t))$, are extracted from tf using the following commands in Python:

```
tf_listener = tf.TransformListener()
(ee_pos, ee_rot) = tf_listener.lookupTransform('/odom',
  '/arm_6_link',rospy.Time.now() - rospy.Duration(delay))
```

where `ee_pos` is the end-effector position, given in Cartesian coordinates; `ee_rot` the rotation, given in quaternions; `'/odom'` a global frame, fixed to the virtual environment; `'/arm_6_link'` a local frame, fixed to link 6 of the virtual robot arm; and `delay` a number between 0 and 1, indicating the time delay.

The reason for measuring link 6 instead of the end-effector itself, is to avoid changing the reference frame when rotating the last joint. The problem of fixing the reference frame for the end-effector movement to link 7, is that the human operator may have difficulty seeing how much the last joint has rotated in total. The last joint angle is measured as the rotation from frame `'/arm_6_link'` to frame `'/arm_7_link'`.

The orientation of the end-effector in quaternions is converted to a rotational matrix, using a function called `quaternion_matrix()`, from a provided *tf.transformations* library. The starting point for the end-effector position, $\mathbf{p}_g(t_0 - T(t_0))$, and orientation, $\theta_g(t_0 - T(t_0))$, is measured when the blue button on the Phantom Omni is pressed, and kept the same until the button is released.

The orientation of the joystick is calculated as a set of Euler-angles, based on the measured joystick configuration, $\mathbf{q}_m$, while the position is given directly, both received from the omni-node by subscribing to the corresponding topic.

The desired velocity is given relative to the orientation of the end-effector, and published to the corresponding topic. This velocity is used by the slave controller to calculate the desired joint velocity, and will be the next topic.

6.3 Slave Controller

This section presents the part of the developed program that is based on the slave controller, designed in Section 4.1. The part is made of the *joystick_idk*- and *inverse_differential_kinematics*-node, and calculates the desired control inputs from a desired end-effector velocity and the configuration of the virtual mobile manipulator.

The calculations of the desired joint velocities, $\mathbf{u}^*$, are found in the node called inverse_differential_kinematics. As seen in Figure 6.1, the communication to the slave controller goes through the joystick_idk-node, except from inverse_differential_kinematics subscribing to the topic with the slave configuration.

The joystick_idk-node designed in this study, is based on a previous version of this node and will be the next topic.

6.3.1 joystick_idk-node

The desired end-effector velocity relative to the end-effector frame is subscribed to by the joystick_idk, where the orientation of the frame is derived in Section 3.5.1.

This node is altered from only accepting a desired linear end-effector velocity to accept both linear and angular velocity, as well as to handle rotation of the end-effector frame. The node is also altered from only being able to specify a desired joint velocity for the robot arm, to also specify the desired velocity for the mobile base.

The desired control inputs are published and made available to the simulator in joystick_idk, but the calculation of the control input is done in the inverse_differential_kinematics-node. The desired end-effector velocities are sent to and the desired control inputs received from inverse_differential_kinematics using *service*, which implies that the inverse_differential_kinematics are called in a similar way as a function.

6.3.2 inverse_differential_kinematics-node

The inverse_differential_kinematics-node is designed to calculate the desired control input from the desired end-effector velocity and configuration of the virtual mobile manipulator. The calculations are based on the slave controller, presented in Section 4.1, and the kinematics for the mobile manipulator, derived in Section 3.5.4.

In addition to the calculating the desired control input, the node includes decision criteria for enabling and disabling movement of the mobile base

The first step, in order to find the desired joint velocities, is to acquire knowledge about the slave configuration, represented by the generalized coordinates $\mathbf{q}_s$. This is done by subscribing to the topic to where the simulator publishes the slave configuration. The values measured from the simulator are updated asynchronously, so the measurements are copied to a local variable when inverse_differential_kinematics is requested to do a calculation for the joystick_idk-node, and kept constant through the calculation. The joint angels of the slave robot arm and the position of the mobile base are extracted directly, while the orientation of the mobile base is measured in quaternions. Quaternions are used to describe a rotation between two frames, as discussed in Section 6.1.1, and defined as a rotation α around a unit vector $\hat{\beta}$, represented as

$$\mathbf{q} = [x,\, y,\, z,\, w]^T = \begin{bmatrix} \hat{\beta}\sin(\frac{\alpha}{2}) \\ \cos(\frac{\alpha}{2}) \end{bmatrix}. \tag{6.1}$$

Assuming that the rotation of the mobile base is only given around the z-axis,

implying that $\hat{\beta} = [0,\ 0,\ 1]^T$, the angle, α, is given as

$$\alpha = 2 \cdot \text{atan2}(z, w). \tag{6.2}$$

The calculation of the desired control input starts after a request from joystick_idk, where the desired end-effector velocity is included. Based on the slave configuration the transformation matrix T_g^n, derived in Section 3.5.1, is found and used to transform the desired end-effector velocity from the end-effector frame to the global frame, $\dot{\xi}^*$.

To prevent movement when the button on the joystick is not pressed, a test is conducted to see if the desired end-effector velocity is zero, assuming that a human operator is not able to keep the joystick completely stationary. If the velocity is zero, the desired control input is set to zero, if not, the manipulability of the slave robot arm is calculated, where the manipulability is found as (4.6), described in Section 4.1.2, with $\alpha = 0$.

To avoid singularity of the slave robot arm, which happens when the angle of joint 2, 4 or 6 of the arm is zero, the corresponding control inputs are set to an arbitrary negative value when the manipulability approaches zero. If the base is not already moving, the manipulability has to be lower than δ_l, for the base to be able to move, but if the base was moving last calculation, the manipulability has to exceed δ_h for the base to stop. δ_l and δ_h were found by altering their values until a desired behavior was seen, giving δ_l and δ_h as 0.036 and 0.042, respectively.

The control input is defined as

$$\mathbf{u} = [v \quad \omega \quad \dot{q}_1 \quad \dot{q}_2 \quad \dot{q}_3 \quad \dot{q}_4 \quad \dot{q}_5 \quad \dot{q}_6 \quad \dot{q}_7]^T \tag{6.3}$$

where $\dot{q}_i$ are the joint velocities, v the linear velocity of the base, and ω the angular velocity of the base. From Section 4.1, the desired control input found as (4.12), that is,

$$\mathbf{u}^* = \bar{\mathbf{J}}_l^\dagger(\mathbf{q}_s)\dot{\mathbf{t}}_l + \mathbf{N}_l(\bar{\mathbf{J}}_e^\dagger(\mathbf{q}_s)\dot{\xi}^* + \mathbf{N}_e(\mathbf{w}_1 + \mathbf{w}_2)), \tag{6.4}$$

where the Jacobian for the joint limits, $\bar{\mathbf{J}}_l$, is defined by (4.1), the velocity task for the joint limits, $\dot{\mathbf{t}}_l$, by (4.3), and the matrix spanning the null space of $\bar{\mathbf{J}}_i$ by $\mathbf{N}_i = \mathbf{I}_7 - \bar{\mathbf{J}}_i^{\dagger}(\mathbf{q}_s)\bar{\mathbf{J}}_i(\mathbf{q}_s)$. $\dot{\xi}^*$ is the desired end-effector velocity, while $\mathbf{w}_1$ and $\mathbf{w}_2$ are optimization criteria, given by (4.10) and (4.11), respectively.

The Jacobian, $\mathbf{J}$, which maps the generalized velocity of the virtual mobile manipulator to the end-effector velocity, is found as derived in Section 3.2. The columns of the Jacobian are calculated as (3.27) for prismatic joints, and as (3.28) for revolute joints. If the manipulability indicates that the base should move, the Jacobian is combined with the nonholonomic constraints, $\mathbf{M}$, derived in Section 3.5.4. The matrix $\mathbf{M}$ is calculated by (3.56), giving

$$\bar{\mathbf{J}}_e(\mathbf{q}) = \mathbf{J}(\mathbf{q})\mathbf{M}(\mathbf{q}). \tag{6.5}$$

If, on the other hand, the base is set to be fixed, the matrix $\bar{\mathbf{J}}_e$ is found by (6.5) as well, but with the two leftmost columns equal zero. These columns are multiplied with the linear and angular velocity of the base, implying $v = \omega = 0$.

The matrix $\bar{\mathbf{J}}_e$ is inverted using singular value decomposition (SVD), with functions provided by the Eigen/SVD-library [127]. A more detailed implementation can be found in Section C.1 seen in

The manipulability gradient, $\frac{\partial \mathcal{P}(\mathbf{q}_s)}{\partial \mathbf{q}_s}$, used in the calculation of $\mathbf{w}_1$, is found numerically. The element i of the gradient is given as

$$\frac{\mathcal{P}(\mathbf{q}_{s,i} + \epsilon) - \mathcal{P}(\mathbf{q}_{s,i})}{\epsilon}. \tag{6.6}$$

where ϵ is set as 0.1. Here $\mathbf{q}_{s,i}$ is the i-th element of the generalized coordinates for the slave robot.

After calculating the desired joint velocities, as well as the desired velocity for the mobile base, the desired control input is sent to joystick_idk as a reply. The velocity limits are verified in joystick_idk, before it is forwarded to the simulator.

The program presented in this chapter, as well as the framework described in Chapter 5, is tested by several experiments. A short description of each test and the results are presented in the next chapter.

Chapter 7

Experiments and Results

To analyze the controllers, proposed in Chapter 3, the framework, described in Chapter 5, and the implementation of the ROS-program, presented in Chapter 6, several experiments are designed. The purpose of these experiments is to test the desired functionalities, including stability, handling of time delay and the degree of information to the human operator.

First, in Section 7.1, the setup for the experiments is described. Section 7.2 presents the different test cases, where the intension is to isolate the different reactions of the master and slave manipulator, to easier compare them with the desired response. Afterwards, in Section 7.3, the results of the different test cases will be presented, with a short description of each result.

7.1 Experimental Setup

Since the control architecture is used on a virtual robot, the experiments are performed using a physics-engine-based simulator on a personal computer. A short description of the simulator with dynamics is given in Section 6.1.1.

The computer used is a Dell Optiplex 990, with Ubuntu 11.10, Oneiric Ocelot, operating system (OS), the latest version of Ubuntu at the start of this study.

Ubuntu is chosen as the preferred OS for implementing this system, because ROS primarily runs on Unix-based platform.

The only external, additional communication from the PC is to the human operated joystick, which only communicates with other systems through FireWire, also known as IEEE 1394 interface.

To communicate over FireWire, a FireWire Peripheral Component Interconnect (PCI) card is installed on the computer. On older Linux versions, libraw1394 is used to communicate directly between user space and IEEE 1394 buses [128], but on newer Linux versions, a new kernel driver stack is implemented, and libraw1394 is no longer supported. This problem is handled by running a script that first makes a spoof device, such that OpenHaptics can communicate with it, then loads a dummy module, making the OpenHaptics believe that raw1394 is loaded.

The PC runs a ROS program, consisting of several nodes, displays the movement of the mobile manipulator on a monitor, and sends force feedback to the joystick. An overview of the implemented program is given in Chapter 6.

Several experiments are performed with the setup described in this section. An introduction to the designed test cases is presented next.

7.2 Test Cases

This section presents several designed test cases. The cases are aimed at testing the desired properties of the proposed framework and control architecture.

The desired properties are as follows:

- **Stability** The ability for the system to come to rest when the input is kept unchanged, as well as to keep the output bounded for bounded input. Could be measured by the oscillations and overshoot made by the output relative to the reference value.
- **Output synchronization** How well the end-effector tracks the movement of the human operated joystick. In this study, this is measured as the deviation in position.
- **Handling of time delay** The stability of the system when under the influence of time delay. The delay in the communication between the user interface (haptic joystick) and the robot to be controlled can make a system unstable, even if the system initially is stable. The time delay can be constant, variable or have elements of both.
- **Smooth transition** The ability to keep the motion of the end-effector and the force feedback approximately the same just before and after a transition between fixed and moving robot base.
- **Fast response** How fast the end-effector approaches the desired position.
- **Informative feedback** The information value of the feedback for the human operator. Measured as the correspondence between the state of the mobile manipulator and the feedback.
- **Intuitive control** How easy it is for the operator to control the slave robot.
- **High manipulability** The size of the reachable area for the end-effector, when applying a small change in the joint angles. A measurement of how easy it is to realize a desired end-effector velocity.

For the stability analysis, six different scenarios exist. With fixed robot base, the system can be unaffected by time delay, or affected by constant or variable time-delay. The system can be affected by the same effects with movable base. The robot base is set to move only when the manipulability

is under a predefined level. Table 7.1 summarizes the scenarios E_1, E_2, E_3, E_4, E_5 and E_6.

Base \ Delay	None	Constant	Variable
Fixed	E_1	E_2	E_3
Movable	E_4	E_5	E_6

Table 7.1: A summary of the different scenarios for analyzing the stability.

In addition to the stability, it will be interesting to analyze the transition between fixed and movable base, to ensure that no unexpected movement or force feedback occurs, such as jumps or shacking. This behavior could be the result of different optimization criteria for the desired control input in the to states. The sudden change in behavior could also affect the end-effector position, and thus the force feedback.

It is also important that the force feedback reflects the state of the mobile manipulator, and gives the operator additional information to the visual feedback. The force feedback should also make the control of the system more intuitive. Whether the control is intuitive or not is mostly subjective, and difficult to measure quantitatively.

Based on the aforementioned scenarios and desired behavior, seven different tests are designed. The tests are performed with different time-delays, where the controller gains for the desired end-effector velocity, k_3 and k_4, are chosen based on the delay. The gains are described in Section 4.2.1, and given by (4.13) and (4.14), that is,

$$\mathbf{v}_e^* = -k_3 \left(\tilde{\mathbf{p}}_e(t, t_0) - k_1 \tilde{\mathbf{p}}_e^*(t, t_0) \right) \tag{7.1}$$

$$\omega_e^* = -k_4 \left(\tilde{\theta}_e(t, t_0) - k_2 \tilde{\theta}_e^*(t, t_0) \right), \tag{7.2}$$

where $\tilde{\mathbf{p}}_e$ and $\tilde{\mathbf{p}}_e^*$ are the measured and desired change in the end-effector position, while $\tilde{\theta}_e$ and $\tilde{\theta}_e^*$ are the measured and desired change in the end-effector orientation. k_1 and k_2 are scaling factors between the joystick and end-effector movement, t the time, and t_0 the start time. The constants, k_3

and k_4, are directly related to the K_s variable, discussed in Section 4.3, which depends on the maximum time-delay, seen in (4.21).

The first six test cases are conducted with predefined input signals, while the last case is performed with human operators to see how intuitive it is to operate the system. The predefined inputs are oscillating signals generated independently of the force feedback. The different cases are given as

Case 1: No time delay and fixed robot base. Predefined input. Aimed at testing the stability and position tracking for the manipulator, in addition to the force feedback.

Case 2: 0.5 sec constant time-delay and fixed base. Predefined input. Aimed at testing the stability and position tracking for the manipulator, as well as the force feedback, with constant delay.

Case 3: 0.5 sec constant plus 0-0.5 sec variable time-delay with fixed base. Predefined input. Tests the stability and position tracking for the manipulator, and the force feedback, with variable delay.

Case 4: No time delay and movable robot base. Predefined input. Aimed at testing the stability and position tracking for the mobile manipulator, in addition to analyze the transition between fixed and moving base.

Case 5: 0.5 sec constant time-delay and movable base. Predefined input. Aimed at testing the stability and position tracking for the mobile manipulator, and the transition between fixed and moving base, with constant delay.

Case 6: 0.5 sec constant plus 0-0.5 sec variable time-delay with movable base. Predefined input. Aimed at testing the stability and position tracking for the mobile manipulator, as well as to analyze the transition between fixed and moving base, with variable delay.

Case 7: 0.5 sec constant plus 0-0.5 sec variable time-delay with movable robot base. Input from human operators. Tests how intuitive it is to operate and control the system with a haptic joystick.

7.3 Results

In this section the results of the different cases, described in the previous section, are presented.

First in this section, are the results from the cases with fixed robot base, then, in Section 7.3.2, the results from similar tests with movable base are found. The last part presents the results from the simulations with actual human operators.

7.3.1 Fixed Robot Base

First, the system is tested with a fixed base. The following three cases are conducted with a predefined input from the master manipulator. The desired change in end-effector position for the slave manipulator, given in local coordinates relative to the end-effector frame, $\tilde{\mathbf{p}}_e^*$, is chosen as

$$\tilde{\mathbf{p}}_e^* = \begin{cases} [40\sin(\frac{\pi}{5}t), \quad 35(\cos(\frac{\pi}{6}t)-1), \quad 10(1-\cos(\frac{\pi}{6}t))]^T, & 0 \leq t \leq 10 \\ [0, \quad 35(\cos(\frac{\pi}{6}t)-1), \quad 10(1-\cos(\frac{\pi}{6}t))]^T, & 10 < t < 12 \\ [0,0,0]^T, & 12 \leq t \leq 18 \end{cases}$$

The purpose of these tests is to analyze the stability and force feedback. Different controller gains are used at each case, based on the time delay.

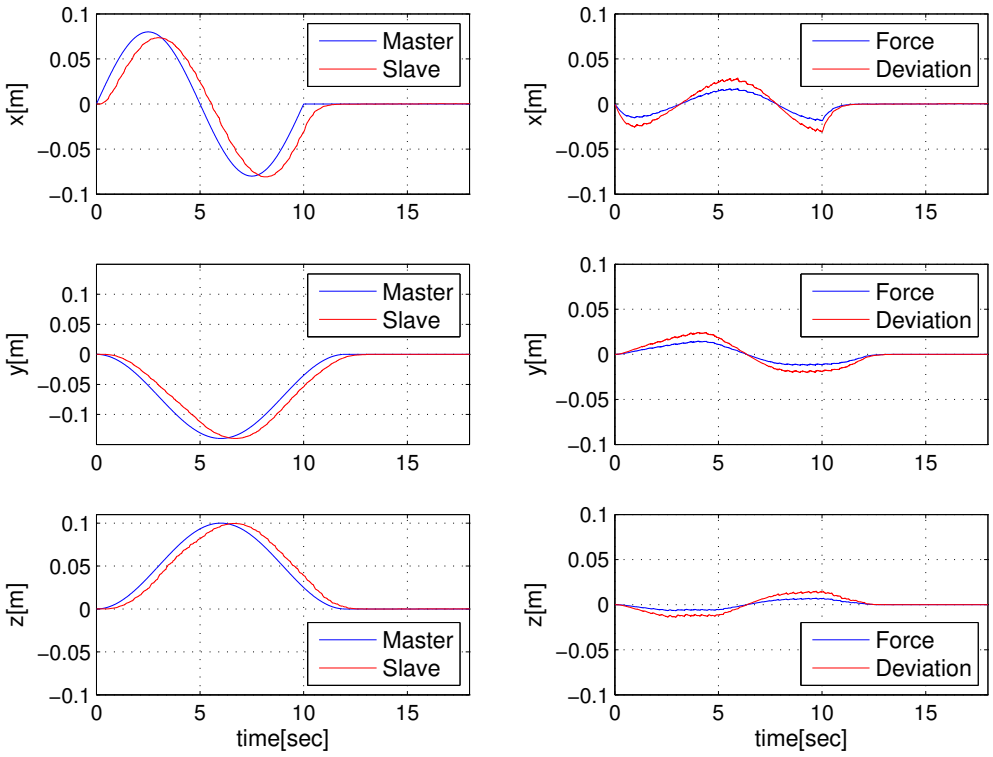
The desired change in the end-effector rotation is set to zero, since the stability is reflected in the position, and the force feedback do not depend on deviation in rotation. The reason for omitting the rotation in the force feedback is that the Phantom Omni is only capable of generating force in 3 DOF, that is, only linear force.

Case 1

In the first test case, the virtual mobile manipulator was controlled by a virtual joystick. The precalculated movement $\tilde{\mathbf{p}}_e^*$ was chosen as input. The time delay was zero, and the controller gains were set as

$$k_3 = 2 \qquad k_4 = 3.$$

The result of the simulation after 18 seconds is shown in Figure 7.1. The figure shows the desired versus simulated change in end-effector position, as well as the force feedback versus the difference between desired and simulated change in end-effector position, all given in Cartesian coordinates. The

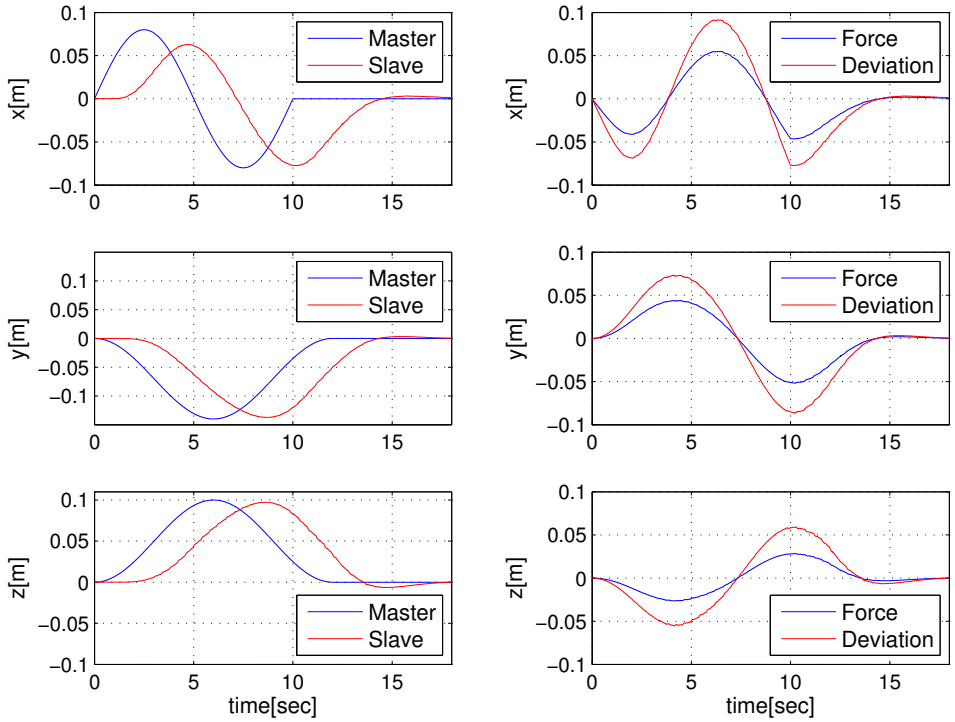


(a) Cartesian position (b) Force feedback and position deviation

Figure 7.1: Simulated master and slave position (a), and position deviation between the two manipulators and force feedback (b), with fixed base and no time-delay.

change in joystick position is multiplied by k_1 to be comparable with the change in end-effector position, while the force is scaled to be comparable with the deviation in position.

The result from this case indicates that the system is able to track the position, without any standard deviation. The response has no overshoots, and the system can be said to be stable. Though, the response is slightly delayed, compared with the input. This delay may be the result of the dynamics for the slave robot, optimization criteria, computation time or low gains.



(a) Cartesian position (b) Force feedback and position deviation

Figure 7.2: Simulated master and slave position (a), and position deviation between the two manipulators and force feedback (b), with fixed base and 1 sec constant time-delay.

Case 2

Again, the virtual mobile manipulator was controlled by a virtual joystick with a predefined motion, but this time affected by a 1 second constant delay. Since the stability is related to both the delay and controller gains, the gains are lowered when the system is influenced by time delay. After analyzing the response in several simulations, the controller gains were set to

$$k_3 = 0.5 \qquad k_4 = 0.75.$$

The result of the test after 18 seconds is shown in Figure 7.2. The figure shows the desired versus simulated change in end-effector position, as well as the force feedback versus the difference between desired and simulated change in end-effector position, all given in Cartesian coordinates. The positions are given in meters, while the force is scaled to be comparable with the difference.

The result from case 2 suggests that the system is stable when effected by constant time-delay. The response has almost no overshoot and has a similar trajectory as the desired input, but with a 2-3 seconds delay. The absence of overshoot is favorable when obstacles are introduced, and important for the stability. The large delay can have been caused by the introduced time-delay itself, or it can be a combination of the new time-delay and lower gains. This could be a problem for the human operator if even greater time-delays are enlarged by the same factor.

Case 3

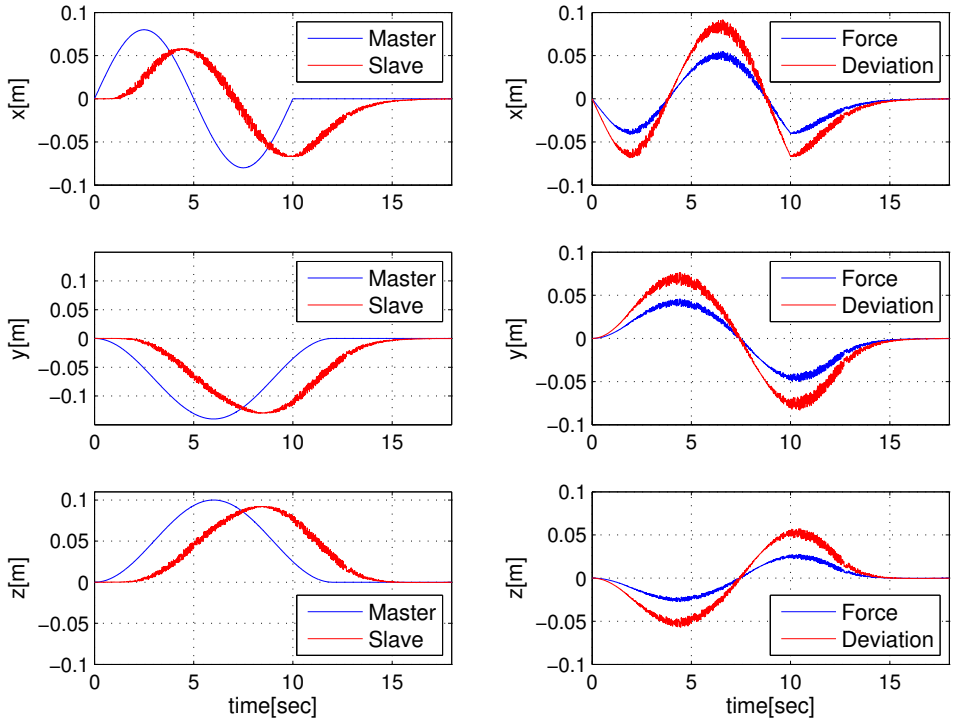
This time, the virtual mobile manipulator was controlled by a virtual haptic device and affected by a 0.5 seconds constant delay in addition to a variable time-delay, randomly chosen between 0 and 0.5 seconds. The controller gains were set to the same as the gains in the previous case, because the maximum

time-delay is equal. The gains were therefore

$$k_3 = 0.5 \qquad k_4 = 0.75.$$

The result of the simulation after 18 seconds is shown in Figure 7.3. The figure shows the desired versus simulated change in end-effector position, as well as the force feedback versus the difference between desired and simulated change in end-effector position, all given in Cartesian coordinates. The positions are given in meters, while the force is scaled to be comparable.

Figure 7.3(a) shows that the system is stable when under influence of variable time delay. The end-effector follows the desired trajectory, but with a damped and delayed movement. The response in Figure 7.3(a) has no over-



(a) Cartesian position (b) Force feedback and position deviation

Figure 7.3: Simulated master and slave position (a), and position deviation between the two manipulators and force feedback (b), with fixed base and 0.5 sec constant plus 0-0.5 sec variable time-delay.

shoot but a high frequency, low amplitude variance. This variance is also seen in Figure 7.3(b) and can cause the joystick to shake and give the human operator a slightly uncomfortable experience.

7.3.2 Moving Robot Base

Next, the system is tested with a movable robot base. The base only moves if the manipulability, given by (4.7), is beneath a given threshold. The following three cases are conducted with a predefined input from the master manipulator as well. The desired change in end-effector position for the slave manipulator, given in local coordinates relative the end-effector frame, $\tilde{\mathbf{p}}_e^*$, is chosen as

$$\tilde{\mathbf{p}}_e^* = \begin{cases} [75\sin(\frac{\pi}{5}t), \quad 75(1-\cos(\frac{\pi}{6}t)), \quad 15(\cos(\frac{\pi}{6}t)-1)]^T, & 0 \leq t \leq 10 \\ [0, \quad 75(1-\cos(\frac{\pi}{6}t)), \quad 15(\cos(\frac{\pi}{6}t)-1)]^T, & 10 < t < 12 \\ [0,0,0]^T, & 12 \leq t \leq 18 \end{cases}$$

Again, the desired change in end-effector rotation is set to zero. For given scaling factors k_1 and k_2, the desired change in end-effector position might be equivalent to a movement of the Phantom Omni that exceeds the physical limits of the joystick. But the results will still be representative for the properties of the system, since the human operator is able to reset the origin of the desired position and generate the same input by combining several shorter movements.

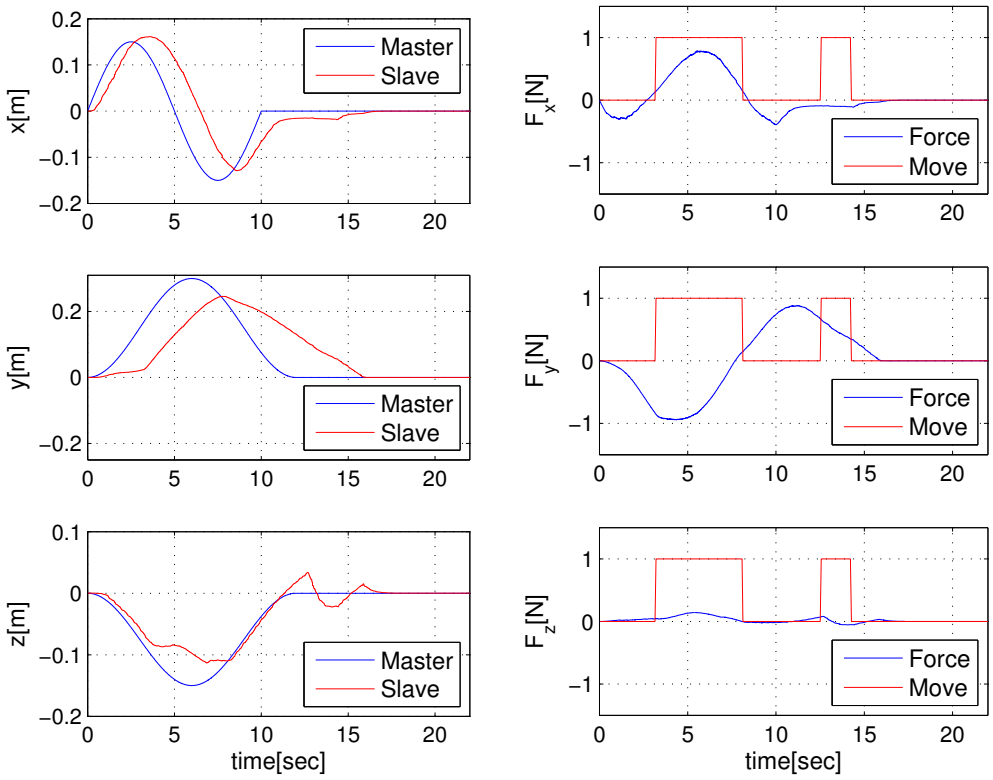
The purpose of these tests is to analyze the stability of the system and the effect of changing state from fixed to movable base, and visa versa. The controller gains are equal to the gains when the base was fixed.

Case 4

With movable base, the virtual mobile manipulator was controlled by a virtual haptic device. The system was not effected by time delay, and the gains were therefore chosen as

$$k_3 = 2 \qquad k_4 = 3.$$

To capture the entire response, the system was simulated for 22 seconds. The left column of Figure 7.4 shows the desired versus the simulated change in end-effector position, partitioned into x-, y- and z-coordinates, while the right column shows the corresponding force feedback and the state indicating



(a) Cartesian position (b) Force feedback and move-state

Figure 7.4: Simulated master and slave position (a), and force feedback together with state indicating movement of the base (b). Tested with movable base and no time-delay.

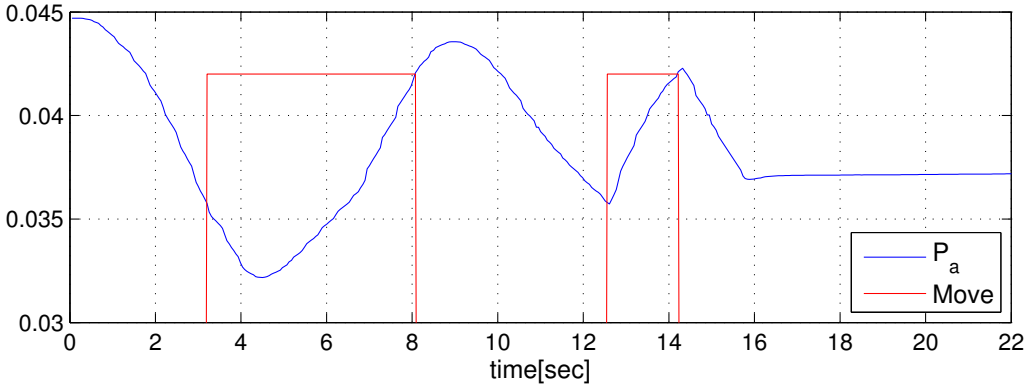


Figure 7.5: Manipulability and state indicating movement of the base.

if the base moves (=1) or not (=0).

Figure 7.5 shows the manipulability, $\mathcal{P}_a(\mathbf{q}_{s,a})$, given by the configuration of the arm, represented by the generalized coordinates $\mathbf{q}_{s,a}$, and the state indicating if the base moves. The mobile base is set to move if the manipulability is lower than 0.036, and stop moving if the manipulability exceeds 0.042. The move-indicator is initially 1 if the base moves and 0 otherwise, but scaled by 0.042 in the figure to easier compare.

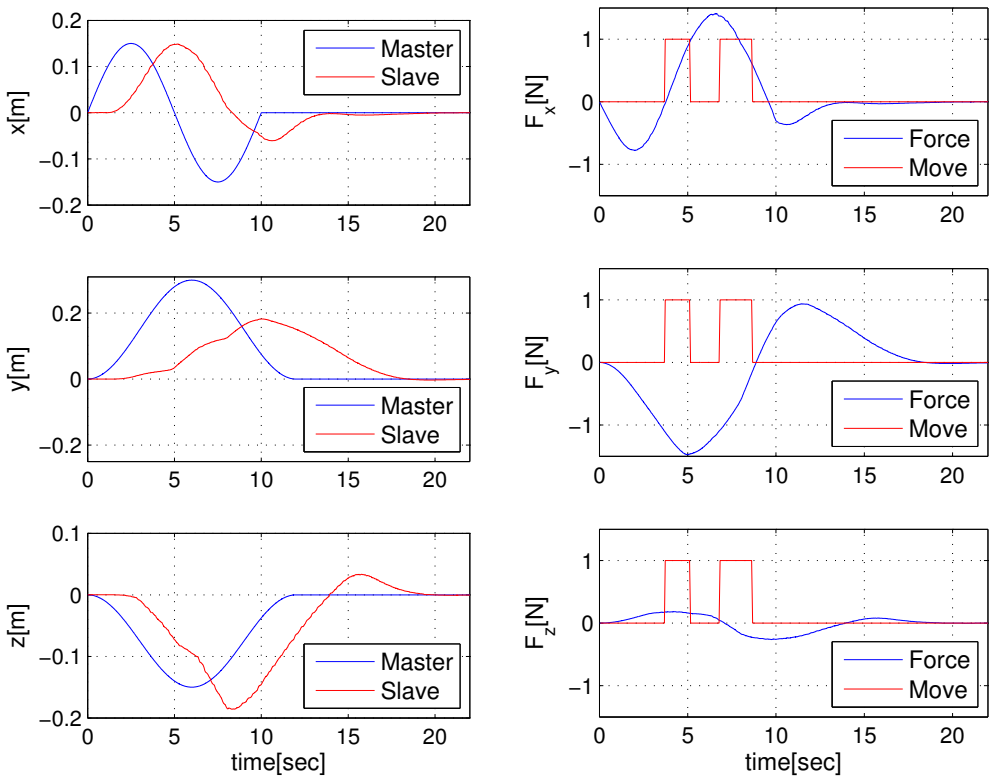
The results from case 4 indicate that the system is stable when the robot base is able to move. The end-effector position settles at the desired stationary position and is quite smooth at the transitions, but has some small overshoots and is rather slow, compared to the input signal.

Case 5

This time, the virtual mobile manipulator was controlled by a virtual haptic device, under the influence of a 1 second constant time-delay. The base was able to move when the manipulability was low enough, and the gains, based on the time delay, were chosen as

$$k_3 = 0.5 \qquad k_4 = 0.75.$$

The system was again simulated for 22 seconds, with the results presented in Figure 7.6 and Figure 7.7. The left column of Figure 7.6 shows the



(a) Cartesian position (b) Force feedback and move-state

Figure 7.6: Simulated master and slave position (a), and force feedback together with state indicating movement of the base (b). Tested with movable base and 1 sec constant time-delay.

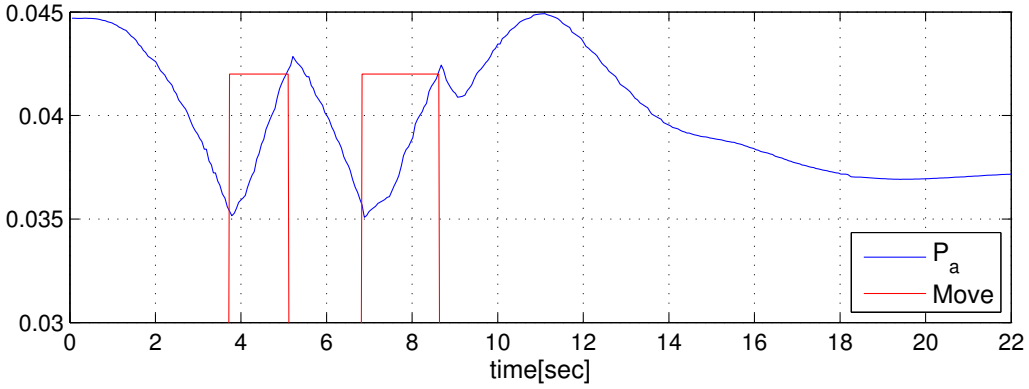


Figure 7.7: Manipulability and state indicating movement of the base.

desired versus the simulated change in end-effector position, partitioned into x-, y- and z-coordinates, while the right column shows the corresponding force feedback and the state indicating if the base moves or not.

Figure 7.7 shows the manipulability, $\mathcal{P}_a(\mathbf{q}_{s,a})$, and the state indicating if the base moves. The move-indicator is scaled by 0.042 in the figure to easier compare with the manipulability.

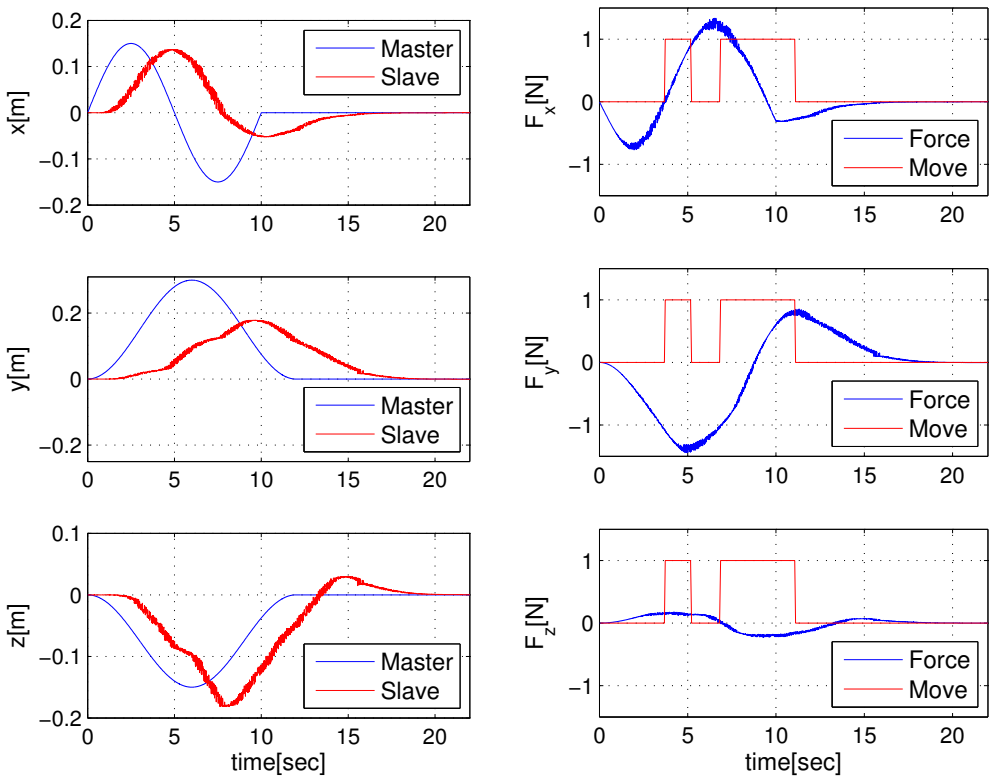
Based on the results from this test case the system seems to be stable when affected by a constant time-delay, and with a movable robot base. The end-effector approaches the desired stationary position, and has a smooth movement at the transition between fixed and moving base. The response of the end-effector position has some overshoot and is quite slow, where the overshoot can create problems in terms of obstacle avoidance.

Case 6

The last experiment with a virtual haptic device and movable base, was simulated with a 0.5 seconds constant time-delay, in addition to a variable time-delay between 0 and 0.5 seconds. The maximum delay is 1 second, so the gains were chosen as

$$k_3 = 0.5 \qquad k_4 = 0.75.$$

Figure 7.8 and Figure 7.9 shows the results after simulating the movement of the virtual mobile manipulator for 22 seconds. The desired versus the simu-



(a) Cartesian position (b) Force feedback and move-state

Figure 7.8: Simulated master and slave position (a), and force feedback together with state indicating movement of the base (b). Tested with movable base and 0.5 sec constant plus 0-0.5 sec variable time-delay.

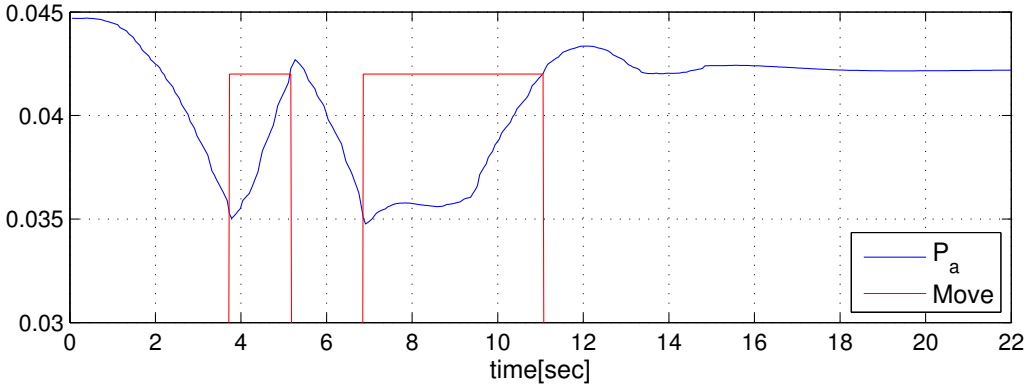


Figure 7.9: Manipulability and state indicating movement of the base.

lated change in end-effector position, partitioned into x-, y- and z-coordinates are presented in the left column of Figure 7.8, while the corresponding force feedback and the indicator for fixed or moving base are seen in the right column.

The manipulability, as well as the state indicating if the base moves, are shown in Figure 7.9, where the base moves if the indicator is 0.042, and fixed if 0.

The results from case 6 suggest that the system is stable, with movable base and under influence of variable time-delay. The mobile manipulator manages to keep a high manipulability, and has only small overshoots in the end-effector position compared with the desired response. A transition between fixed and movable base do not create any abrupt change in the end-effector position. However, a high frequency, low amplitude variance can be seen in both the end-effector position and force feedback. Though the simulations are performed based on the dynamics of a real robot, the vibrations in the virtual end-effector position might cause problem in a real life application.

7.3.3 Human Operated Control

In the last case, the virtual mobile manipulator is controlled by a real human operator. The system is tested with a 0.5 seconds constant time delay, in addition to a variable time-delay between 0 and 0.5 seconds, and with movable base.

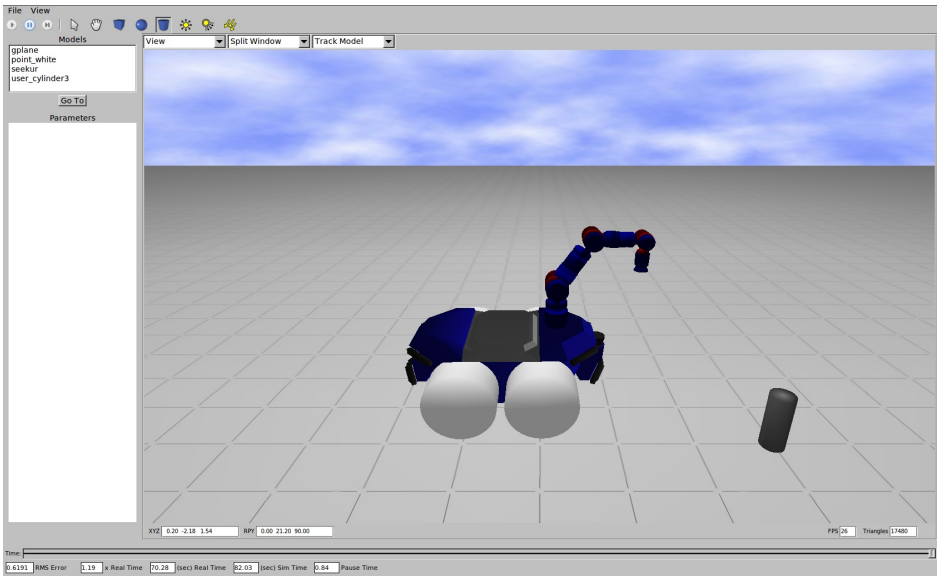
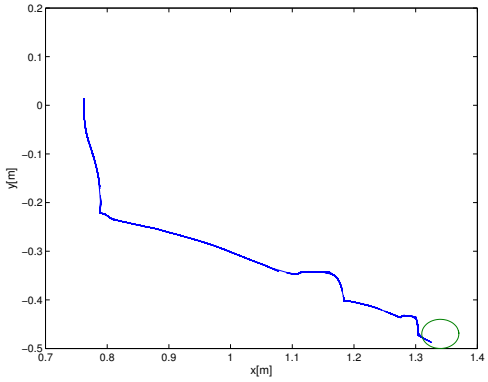


Figure 7.10: Virtual mobile manipulator in initial position.

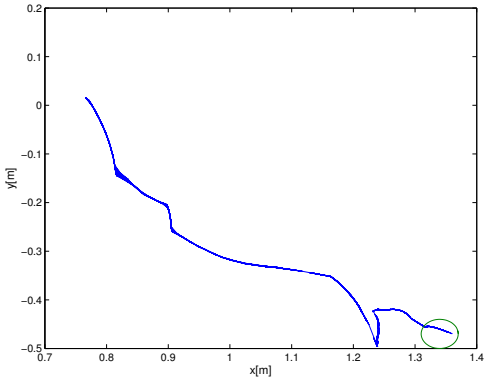
The purpose of this test is to analyze how intuitive it is to operate the joystick and control the mobile manipulator, in addition to see the response of the system, when interacting with a human operator.

Three different operators were asked to move the end-effector of the virtual mobile manipulator from an initial position to the top of a cylinder, located diagonally in front of the slave manipulator. The operators are students, studying at the Department of Engineering Cybernetics, NTNU. The initial position and cylinder are shown in Figure 7.10.

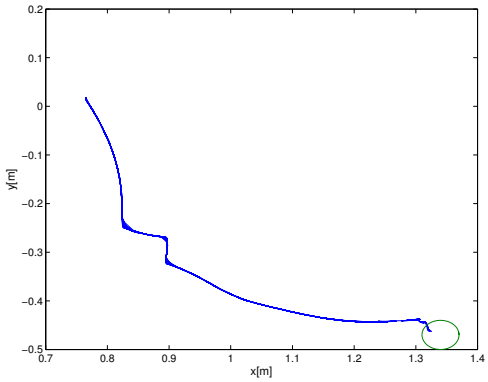
The human operators 1, 2 and 3, completed the task in respectively 24, 25 and 18 seconds. Here operator 1 and 2 are fairly new in operating the virtual manipulator, while operator 3 has some more experience. The horizontal



(a) Human operator 1.



(b) Human operator 2.



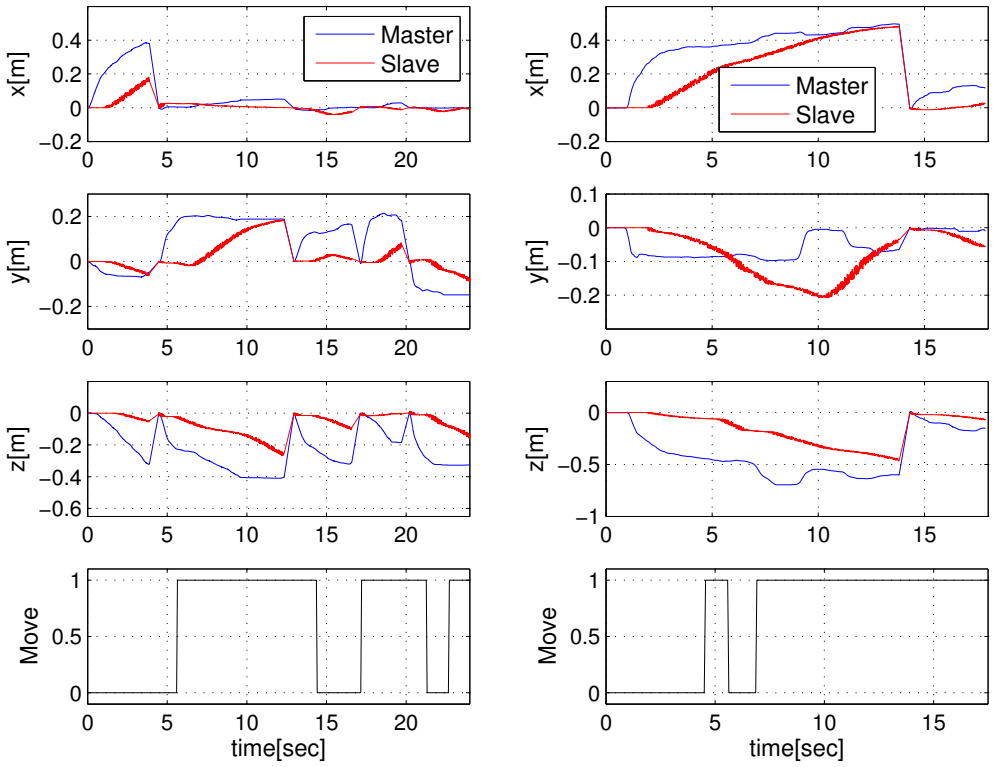
(c) Human operator 3.

Figure 7.11: The position of the virtual slave end-effector in the horizontal plane, controlled by human operator 1 (a), 2 (b) and 3 (c).

movement of the virtual end-effector are presented in Figure 7.11, where the green circle indicates the position of the cylinder in the test.

Figure 7.12 shows the position of the master and slave manipulator, in Cartesian coordinates relative to the end-effector, as well as the state indicating movement of the base. The left column contains the results from operator 1, while the right column contains the results from operator 3. The figure are presented to compare the response of the system when controlled by an inexperienced operator and when controlled by an operator who is more familiar to controlling the system.

By comparing the trajectories, created by operator 1 and 3, with the relative



(a) Operator 1 has no experience.

(b) Operator 3 has some experience.

Figure 7.12: Master and simulated slave position, controlled by an inexperienced operator (a) and an operator with more experience (b) in controlling the mobile manipulator using the haptic joystick.

joystick and end-effector position, shown in Figure 7.12, one can see how the input from the joystick correlates with the movement of the end-effector. The trajectories seems to be partitioned into parts divided by a sudden change in direction, which could be caused by the transition between fixed and movable base or intentional change in the direction of the desired motion. If the change is caused by the operator trying to correct former movement of the joystick, it could indicate that the operation is not intuitive.

As seen in Figure 7.11, all three operators are able to place the end-effector at the desired point, marked by the cylinder shown in Figure 7.10. The trajectories deviate somewhat from a straight line, which could be an issue when introducing obstacles.

The position of the joystick and end-effector, seen in Figure 7.11, shows that operator 1 changes the starting point often, in addition to performing large movement with the joystick. This could be tiring, and prevent an accurate control of the end-effector position. When controlled by a human operator the mobile manipulator do not produce any sudden change in the end-effector position at the transition from fixed to moving robot base.

The results presented in this section are further discussed and analyzed in the next chapter.

Chapter 8

Discussion

In this chapter the results from the previous chapter are analyzed and discussed, with the intention of comparing the properties of the designed control architecture and framework with the desired behavior.

The desired properties are described in more detail in Section 7.2 and given as follows:

- Stability
- Output synchronization
- Handling of time delay
- Smooth transition, from fixed to moving base
- Fast response
- Informative feedback
- Intuitive control
- High manipulability

The rest of this chapter is partitioned in two parts, where the first part discusses the results reflecting the properties of the slave controller. The analysis of the results describing the performance of the master controller, is presented in the last part.

8.1 Slave Controller

In this study, the term slave controller is used to describe the calculation of a desired control input, $\mathbf{u}^*$, based on a desired end-effector velocity in operational space. For the robot arm, the control input corresponds to joint velocities, while for the robot base, this input corresponds to a linear and angular velocity. To isolate the performance of the slave controller, the inputs to the first six cases, presented in Chapter 7, are predefined and independent of the movement of the human operated joystick.

The basic stability and output synchronization were tested in the first three experiments. The simulations were performed by using a precalculated, oscillating movement as input, and by enforcing a fixed robot base. As seen in Figure 7.1(a), Figure 7.2(a), and Figure 7.3(a), the system is stable and eliminates any standard deviation, even under influence of time delay.

Without communication delay, the movement of the end-effector shows no sign of overshoot, tracks the desired trajectory well, and has a fast response. When affected by constant time-delay, the end-effector movement has a larger deviation from the reference, compared to the case without time-delay, and has a small overshoot. The response is rather slow, even when subtracting the 1 second delay from the end-effector position. The movement of the end-effector is more damped, and has an additional low amplitude, high frequency response when under influence of variable time-delay, compared to the response with constant delay. Since the variable time-delay is chosen randomly and discrete, the introduced delay could reflect a communication network with packet loss. The output synchronization is approximately the same for constant and variable time-delay. Except for the additional end-effector vibration, the results indicate that the system can handle constant and variable time delay with good, but delayed, tracking ability. The response is fast when unaffected by time delay, but decreases when communication delay is introduced.

Test case 4, 5 and 6, are performed to test the stability and handling of time

delay, when the robot base is able to move. The transition between fixed and movable base is also analyzed, as well as the tracking ability and manipulability. Again, the simulations were performed by using a precalculated, oscillating movement as input. The input had higher amplitude than the three previous experiments.

Figure 7.4(a) shows the desired and measured end-effector position in Cartesian coordinates when the system is unaffected by time delay and is able to move the robot base. Compared to case 1, the end-effector position has a larger overshoot, slower response, and a more stuttering movement. The error and stuttering are greatest when the base moves. The slow response is assumed to be a result of larger amplitude in the desired input, while the stuttering can be caused by different calculation of the applied force for the mobile base and the arm joints or by different optimization criteria. The result also suggests that the system is less stable with movable base, though this could be caused by larger movement in the desired position as well.

The movement of the end-effector in case 5 are shown in Figure 7.6(a). The response is slower and the overshoot in the z-direction is larger than in the previous case, while the amplitude of the response is smaller in x- and y-direction. The movement of the end-effector is also more smooth compared to the previous case. This could be a result of lower gains, and therefore smaller difference in applied force at the transition between moving and fixed base.

The end-effector position from the last experiment with fixed input is shown in Figure 7.8(a). In this case the time-delay was varying between 0.5 and 1 second, and the base moved when the manipulability was lower than a given threshold. As with fixed base, the introduction of variable time-delay, gives a noisy-like response, probably caused by packet loss, with a smaller amplitude and delay. The results indicate that the system is more stable, when considering overshoot and delay, with variable than constant time-delay.

The manipulability when the system is not affected by time delay, shown in

Figure 7.5, suggest that the controller is able to maintain a manipulability over a certain level. Most of the increase occurs when the robot base is able to move. The manipulability shown in Figure 7.7, indicates that optimization of the manipulability is faster when influenced by time delay, particularly when the robot base moves. This is probably caused by lower amplification of the desired velocity, compared with the weighting of the optimization of the manipulability. From Figure 7.9, the manipulability optimization does not seem as dominant with variable time-delay as with constant, and the mobile manipulator uses more time in a state where the robot base moves.

All the cases where the robot base is able to move, show that the end-effector position is not severely affected by the transition from fixed to moving base, or visa versa. The effect on the end-effector position is more significant with higher gains and without time delay.

In addition to the analysis of the controller for the virtual mobile manipulator, it is necessary to discuss the value of the feedback information and how intuitive the operation of the system is. Both the feedback and desired end-effector velocity are calculated by, what in this thesis is called, master controller and discussed in further detail next.

8.2 Master Controller

The term master controller is used to describe the calculation of force feedback to the human operator, and interpretation of the joystick movement as a desired end-effector velocity. The force is given in three dimension in the linear directions, while both the position and orientation of the joystick are measured. A precalculated joystick trajectory is used to analyze the force feedback quantitatively, while the system is controlled by human operators to get a more subjective and qualitative analysis of the force feedback.

As seen in Figure 7.1(b), the position error is small when the movement of the joystick is slow and without time delay. This is reflected in a small force in all three dimensions. When the system is introduced to a constant time-delay,

shown in Figure 7.2(b), the deviation between desired and measured end-effector position increases and the force gets larger. The last experiment, seen in Figure 7.3(b), is conducted with a variable time-delay. The force feedback is a good reflection of the position error, but has an additional, high frequency, variation with small amplitude.

The experiments suggest that the force feedback gives the human operator a good indicator of the deviation in the position between desired and simulated end-effector position. However, the high frequency variation, introduced to the feedback when the system is affected by variable time-delay, may cause the joystick to shake and give the human operator a slightly uncomfortable experience.

When the base is able to move, it is important that the transitions between movable and fixed robot base do not affect the force feedback significantly. Without time delay, seen in Figure 7.4(b), the force is smooth both from movable to fixed base and visa versa. The results from the experiments with constant and variable time-delay, shown in Figure 7.6(b) and Figure 7.8(b) respectively, indicate smooth transitions as well.

Even though the quantitative measurements suggest a desired behavior, it is important that the human operator is able to make sense of the feedback and move the mobile manipulator in a desired direction. This is a more subjective part of the analysis, and is performed by having three people operating the system individually. The goal is to move the end-effector of the virtual mobile manipulator from an initial position to the top of a cylinder, seen in Figure 7.10.

Figure 7.11 shows the end-effector trajectories for the virtual mobile manipulator, when controlled by the human operators and affected by a variable time-delay. The trajectories are projected onto the x-y plane, with the circle indicating the position of the cylinder. The results demonstrate that the operators are able to place the end-effector at the correct position, without deviating too much from the optimal path.

Operator 1 and 3 represent an inexperienced and a more experienced opera-

tor, respectively. By comparing the trajectories, created by operator 1 and 3, with the relative joystick and end-effector position, shown in Figure 7.12, one can see how the input from the joystick correlates with the movement of the end-effector. The trajectories seem to be partitioned into parts divided by a sudden change in direction. This change could be caused by the transition between fixed and movable base, or intentional change in the direction of the desired motion. If the change is caused by the operator trying to correct the former movement of the joystick, it indicates that the operation is not intuitive.

One choice of design that might confuse the operator is that the relative change in position and orientation is measured in the end-effector frame, while the graphical representation shows the virtual mobile manipulator from an angle fixed to the environment, as in Figure 7.10. This could complicate the translation from the motion the operator desires to the appropriate applied force by the human operator, necessary to achieve this movement.

Another potential issue for a human operator, is the time delay. Figure 7.12(a) shows how an inexperienced operator moves the joystick, starting a new point of reference four times throughout the simulation. The operator changes the desired position faster than the slave robot can follow, and in most cases interrupts the motion before the end-effector reaches the desired position. The input from a somewhat more experienced operator is seen in Figure 7.12(b), and shows that it is possible to apply the desired input with only one new starting point, and by using shorter time. In both experiments, the speed of the master manipulator is somewhat larger than the slave manipulator.

The desired change in position, presented in Figure 7.12, shows that the potential shaking of the joystick, mentioned above, vanishes when the system is operated by a person, even when affected by variable time-delay. However, the vibration of the virtual end-effector still occurs.

The most important results from the discussion in this chapter and the overall conclusion of this study are presented in the next chapter, along with potential improvements and further work.

Chapter 9

Conclusion and Further Work

This chapter presents the conclusion of the discussion in the previous chapter and the work done in this study, as well as aspects of the results that may be improved.

The conclusion can be found in Section 9.1, while Section 9.2 lists possible improvements by the design and suggestions for further work.

9.1 Conclusion

In this thesis a stable control architecture for a bilateral teleoperation system has been developed. A haptic joystick was used to control the end-effector of a mobile manipulator in 6 DOF, that is, both position and orientation. The architecture was developed for a system consisting of a Phantom Omni joystick and a Schunk LWA3 7-DOF manipulator, mounted on a Seekur Jr. wheeled mobile base.

The control scheme calculates a desired end-effector velocity for the mobile manipulator and a force feedback to the human operator, based on the con-

figuration of the master and slave manipulator. The force feedback depends on the position error between the desired and measured end-effector position, as well as the linear velocity of the joystick. As a basis for finding the desired joint velocities for the mobile manipulator, the reference velocity for the end-effector is used. The calculation of the joint velocities ensures compliance with the nonholonomic constraints, introduced by the wheeled base. Several tasks are attempted met, and organized in a hierarchy. First, in the hierarchy, is the enforcing of joint limits, second, is ensuring the desired end-effector velocity, before optimizing the manipulability. To increase the accuracy of the end-effector motion, the joystick movement is scaled down before calculating the desired velocity. The mobile robot base is kept fixed as long as the manipulability is over a given threshold, also to increase the accuracy. When the manipulability drops under a certain level the base begins to move, with the purpose of increasing the manipulability.

The control architecture was implemented using ROS, and tested with a physics-engine-based simulator with a model of the mobile manipulator. The results from several tests show that the system with the proposed architecture is able to handle both constant and time-varying communication delay, while ensuring stability. The end-effector of the mobile manipulator is able to track the desired position and eliminate any standard deviation, but with delayed reaction when communication delay is introduced. After a period with moving robot base, the manipulability increases, which ensures a high level of manipulability during the operation. The force feedback reflects the measured position error without vibration when operated by a human operator, and is not directly affected by the transition between fixed and movable base. Real human operators were able to move the end-effector of the mobile manipulator to a desired position, without deviating to much from a straight path, indicating an intuitive control.

However, the introduction of time delay requires lower controller gains, which leads to a slower response of the mobile manipulator in addition to the delay already caused directly by the communication delay. Human operators experience that the desired position and orientation, based on the motion applied

to the joystick, deviate much from the visualization when introduced to large time-delays. This perception is reflected by the high number of new starting points throughout the operations. In addition, the operators claim to have difficulty visualizing the position of the virtual end-effector in all dimensions on the 2D-screen.

Though most of the behaviors are as desired, several aspects can be improved. Potential improvements and suggestions for further work are presented in the next section.

9.2 Further Work

The designed system consists of many elements, which are optimized in various degree. Most parts can either be further optimized or even changed to enhance the velocity of the response, stability and degree of intuitive control for the system.

With the system affected by time delay, it is shown that a more experienced operator is able to move the end-effector faster and with fewer starting points than an unexperienced operator. The use of a skilled operator increases the speed and precision of an operation, though it will not affect the properties of the designed system.

Among the desired improvements is the handling of time delay regarding the presentation of the mobile manipulator configuration to the human operator. A solution is to run a simulation without time delay in parallel with the delayed system, and present the predicted motion to the human operator. However, this predicted model would be exact for a virtual robot, and therefore only represent the operation without delay. For a bilateral teleoperation system with a physical slave manipulator, the error between the model and real slave can be adjusted by comparing the simulated with the measured movement.

Human operators have some difficulties relating the two dimensional, graphical representation to three dimensional position of the slave end-effector. This could be improved by mounting an extra camera, or point of view, on the end-effector of the slave robot arm. The human interaction might also be improved by introducing an additional interpretation of the joystick movement. For instance, by giving a desired motion relative the global frame when the camera is fixed to the environment, relative link 0 if the camera is fixed to the base, or relative link 1 if the view is fixed to this link.

When time delay is introduced, the position error increases dramatically. This could be handled by increasing the gains for the force feedback when the system is affected by time delay, or by designing a gain given by the delay. Changing the gains, or using another feedback scheme, may also increase the information value of the feedback.

If the mobile manipulator is placed in an environment with obstacles, it would be beneficial to use force rendering scheme to create a virtual environmental force between the slave robot and any objects. The distance to an obstacle could be measured by a laser or ultrasonic scanner, and used to create a repulsive force through the joystick.

To reach position far from the initial position, the mobile manipulator would use a lot of time and the human operator has to create several starting points. This could be handled, while maintaining a high precision, by either use a button or key press to indicate larger movement or by calculating the desired speed using several gains, one for shorter joystick motions and one for larger. Another solution is to switch the translation of the measured change in joystick position between a desired velocity and change in position for the end-effector.

The constants used in the controller for the mobile manipulator could advantageously be optimized further, particularly with regards to the optimization criteria. One solution could be to use an integration term as weighting for the manipulability when the base is moving to speed up the optimization in addition to get a smoother transition between fixed and movable base.

The dynamics for the mobile base, used in the physics-engine-based simulator are not entirely convincing, and it might be necessary to include a more realistic model for the robot base. It would also be interesting to use another controller or different gains in the calculation of the torques for the robot arm, for instance an adaptive controller.

Bibliography

[1] *The American Heritage Dictionary of the English Language*. 4th. Houghton Mifflin Company, 2009.

[2] T. B. Sheridan. "Telerobotics, Automation, and Human Supervisory Control". In: *MIT Press*. Cambridge, MA, 1992.

[3] R. Murphy and E. Rogers. "Cooperative assistance for remote robot supervision". In: *Presence* 5.2 (1996), pp. 224–240.

[4] QinetiQ. URL: `http://www.qinetiq.com/`.

[5] iRobot. URL: `http://www.irobot.com/`.

[6] K. Nonami and N. Shimoi. "Development of Teleoperated Six-legged Walking Robot for Mine Detection and Mapping of Mine Field". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 1. 2000, pp. 775–779.

[7] F. Smith, D. Backman, and S. Jacobsen. "Telerobotic maipulator for hazardous environments". In: *Journal of Robotic Systems* 9.2 (1992), pp. 251–260.

[8] G. Junyao et al. "Heavy Explosive Removing Robot Control Technique Research". In: *Intelligent Human-Machine Systems and Cybernetics, 2009. IHMSC '09. International Conference on*. Vol. 1. 2009, pp. 85 –89. DOI: `10.1109/IHMSC.2009.30`.

[9] K. Kim et al. "Robotic Contamination Cleaning System". In: *IEEE Conference on Intelligent Robots and Systems*. Vol. 2. Lausanne, Switzerland, 2002, pp. 1874–1879.

[10] A. Kwitowski, W. Lewis, and W. Mayercheck. "Computer-based, teleoperation of a new highwall mining system". In: *Robotics and Au-*

tomation, 1989. Proceedings., 1989 IEEE International Conference on. 1989, 1478 –1483 vol.3. DOI: 10.1109/ROBOT.1989.100188.

[11] A. Kwitowski, W. Mayercheck, and A. Brautigam. "Teleoperation for continuous miners and haulage equipment". In: *Industry Applications, IEEE Transactions on* 28.5 (1992), pp. 1118 –1125. ISSN: 0093-9994. DOI: 10.1109/28.158837.

[12] N. Ruangpayoongsak, H. Roth, and J. Chudoba. "Mobile Robots for Search and Rescue". In: *IEEE International Workshop on Safety, Security and Rescue Robotics*. Kobe, Japan, 2005.

[13] P. Ridao et al. "Underwater Telerobotics for Collaborative Research". In: *Advances in Telerobotics* 31 (2007), pp. 347–359.

[14] Q. Lin and C. Kuo. "Virtual Tele-Operation of Underwater Robots". In: *IEEE International Conference on Robotics and Automation*. Vol. 2. Albuquerque, NM, USA, 1997, pp. 1022–1027.

[15] E. Stoll and D. Kwon. "The Benefit of Multimodal Telepresence for In-Space Robotic Assembly". In: *Robotics and Applications* (2009).

[16] K. Pfeiffer, M. Bengel, and A. Bubeck. "Offshore robotics - Survey, implementation, outlook". In: *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. 2011, pp. 241 –246. DOI: 10.1109/IROS.2011.6094661.

[17] H. Surmann et al. "Teleoperated Visual Inspection and Surveillance with Unmanned Ground and Aerial Vehicles". In: *International Journal of Online Engineering (iJOE)* 4.4 (2008), pp. 26–38.

[18] M. M. Lirici, V. Papaspyropoulos, and L. Angelini. "Telerobotics in Medicine and Surgery". In: *Min Invas Ther and Allied Technol* 6.5-6 (1997), pp. 364–378.

[19] I. Belousov et al. "Motion planning for the large space manipulators with complicated dynamics". In: *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*. 2005, pp. 2160 –2166. DOI: 10.1109/IROS.2005.1545547.

[20] J. Vertut and P. Coiffet. *Teleoperations and Robotics: Evolution and Development*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1986.

[21] N. Chopra, R. O. Mark W. Spong, and N. E. Barabanov. "On Tracking Performance in Bilateral Teleoperation". In: *IEEE Transactions on Robotics* 22.4 (2006), pp. 861–866.

[22] P Hokayem and M Spong. "Bilateral teleoperation: An historical survey". In: *Automatica* 42.12 (2006), pp. 2035–2057. URL: http://linkinghub.elsevier.com/retrieve/pii/S0005109806002871.

[23] N. Tesla, L. I. Anderson, and G. Peterson. *Nikola Tesla: Guided Weapons and Computer Technology*. Tesla Presents Series. Twenty First Century Books (CO), 1998.

[24] T. B. Sheridan. "Telerobotics". In: *Automatica* 25.4 (1989), pp. 487–507.

[25] T. Sheridan and W. Ferrell. "Remote Manipulative Control with Transmission Delay". In: *Human Factors in Electronics, IEEE Transactions on* HFE-4.1 (1963), pp. 25 –29. ISSN: 0096-249X. DOI: 10.1109/THFE.1963.231283.

[26] W. R. Ferrell. "Remote Manipulation with Transmission Delay". In: *IEEE Transactions on Human Factors in Electronics* 6.1 (1965), pp. 24–32.

[27] W. R. Ferrell. "Delayed Force Feedback". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 8.5 (1966), pp. 449–455.

[28] W. R. Ferrell and T. B. Sheridan. "Supervisory control of remote manipulation". In: *Spectrum, IEEE* 4.10 (1967), pp. 81 –88. ISSN: 0018-9235. DOI: 10.1109/MSPEC.1967.5217126.

[29] "Telerobotics". In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Springer, 2008. Chap. 31.

[30] G. Raju, G. Verghese, and T. Sheridan. "Design issues in 2-port network models of bilateral remote manipulation". In: *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*. 1989, 1316 –1321 vol.3. DOI: 10.1109/ROBOT.1989.100162.

[31] B. Hannaford and P. Fiorini. "A detailed model of bi-lateral teleoperation". In: *Systems, Man, and Cybernetics, 1988. Proceedings of the*

1988 IEEE International Conference on. Vol. 1. 1988, pp. 117 –121. DOI: 10.1109/ICSMC.1988.754254.

[32] R. J. Anderson and M. W. Spong. "Bilateral Control of Teleoperators with Time Delay". In: *IEEE Transactions on Automatic Control* 34.5 (1989), pp. 494–501.

[33] G. Niemeyer and J.-J. E. Slotine. "Stable Adaptive Teleoperation". In: *American Control Conference, 1990*. 1990, pp. 1186 –1191.

[34] D. Lawrence. "Stability and transparency in bilateral teleoperation". In: *Decision and Control, 1992., Proceedings of the 31st IEEE Conference on*. 1992, 2649 –2655 vol.3. DOI: 10.1109/CDC.1992.371336.

[35] Y. Yokokohji and T. Yoshikawa. "Bilateral control of master-slave manipulators for ideal kinesthetic coupling-formulation and experiment". In: *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*. 1992, 849 –858 vol.1. DOI: 10.1109/ROBOT.1992.220189.

[36] R. Lozano, N. Chopra, and M. Spong. "Passivation Of Force Reflecting Bilateral Teleoperators With Time Varying Delay". In: *in Proceedings of the 8. Mechatronics Forum*. 2002, pp. 24–26.

[37] G. Niemeyer and J.-J. Slotine. "Using wave variables for system analysis and robot control". In: *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*. Vol. 2. 1997, 1619 – 1625 vol.2. DOI: 10.1109/ROBOT.1997.614372.

[38] N. Chopra, M. W. Spong, and R. Lozano. "Synchronization of bilateral teleoperators with time delay". In: *Automatica* 44.8 (2008), pp. 2142–2148.

[39] E. Nuño, R. Ortega, and L. Basañez. "An adaptive controller for nonlinear teleoperators". In: *Automatica* 46.1 (2010), pp. 155 –159. ISSN: 0005-1098. DOI: 10.1016/j.automatica.2009.10.026. URL: http://www.sciencedirect.com/science/article/pii/S0005109809004877.

[40] I. G. Polushin and H. J. Marquez. "Stabilization of bilaterally controlled teleoperators with communication delay: An ISS approach". In: *International Journal of Control* 76.8 (2003), pp. 858–870. DOI: 10.1080/0020717031000116515. eprint: http://www.tandfonline.

com/doi/pdf/10.1080/00207170310001165l5. URL: http://www.tandfonline.com/doi/abs/10.1080/00207170310001165l5.

[41] N. Diolaiti and C. Melchiorri. "Teleoperation of a mobile robot through haptic feedback". In: *Haptic Virtual Environments and Their Applications, IEEE International Workshop 2002 HAVE*. 2002, pp. 67 –72. DOI: 10.1109/HAVE.2002.1106916.

[42] J. Lim, J. Ko, and J. Lee. "Internet-based teleoperation of a mobile robot with force-reflection". In: *Control Applications, 2003. CCA 2003. Proceedings of 2003 IEEE Conference on*. Vol. 1. 2003, 680 –685 vol.1. DOI: 10.1109/CCA.2003.1223519.

[43] O. J. Rösch, K. Schilling, and H. Roth. "Haptic interfaces for the remote control of mobile robots". In: *Control Engineering Practice* 10.11 (2002), pp. 1309 –1313. URL: http://linkinghub.elsevier.com/retrieve/pii/S0967066102001533.

[44] ROS. URL: http://www.ros.org/wiki/.

[45] B. Skumsnes. "Teleoperation of Mobile Robot Manipulators". Department of Engineering Cybernetics, Norwegian University of Science and Technology, 2011.

[46] O. Khatib. "A unified approach for motion and force control of robot manipulators: The operational space formulation". In: *Robotics and Automation, IEEE Journal of* 3.1 (1987), pp. 43 –53. ISSN: 0882-4967. DOI: 10.1109/JRA.1987.1087068.

[47] O. Khatib. "Dynamic Control of Manipulators in Operational Space". In: *6th CISM-IFToMM Congress on Theory of Machines and Mechanisms*. New York: Wiley, 1983, pp. 1128–1131.

[48] J. Nakanishi et al. "Operational Space Control: A Theoretical and Empirical Comparison". In: *The International Journal of Robotics Research* 27.6 (2008), pp. 737–757. DOI: 10.1177/0278364908091463. eprint: http://ijr.sagepub.com/content/27/6/737.full.pdf+html. URL: http://ijr.sagepub.com/content/27/6/737.abstract.

[49] A De Luca, G Oriolo, and B Siciliano. "Robot Redundancy Resolution at the Acceleration Level". In: *Laboratory Robotics and Automation* 4 (1992), pp. 97–106.

[50] L. Sentis and O. Khatib. "Synthesis of Whole-Body Behaviors through Hierarchical Control of Behavioral Primitives". In: *Robotics and Automation, IEEE Journal of* 2.4 (2005), pp. 505–518.

[51] T. Yoshikawa. *Foundations of Robotics: Analysis and Control*. The MIT Press, 1990.

[52] T. Yoshikawa. "Analysis and Control of Robot Manipulators with Redundancy". In: *Robotics Research The First International Symposium*. Ed. by M Brady and R. Paul. MIT Press, 1984, pp. 735–747.

[53] T. Yoshikawa. "Manipulability of Robotic Mechanisms". In: *The International Journal of Robotics Research* 4.2 (1985), pp. 3–9. URL: http://ijr.sagepub.com/cgi/doi/10.1177/027836498500400201.

[54] B. Bayle, J.-Y. Fourquet, and M. Renaud. "Manipulability analysis for mobile manipulators". In: *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*. Vol. 2. 2001, 1251 –1256 vol.2. DOI: 10.1109/ROBOT.2001.932782.

[55] G. Campion, G. Bastin, and B. Dandrea-Novel. "Structural properties and classification of kinematic and dynamic models of wheeled mobile robots". In: *Robotics and Automation, IEEE Transactions on* 12.1 (1996), pp. 47 –62. ISSN: 1042-296X. DOI: 10.1109/70.481750.

[56] B. Bayle, J. y. Fourquet, and M. Renaud. "Manipulability of wheeled mobile manipulators: application to motion generation". In: *International Journal of Robotics Research* 22 (2003), pp. 565–581.

[57] *Random House Dictionary*. Random House, Inc., 2012.

[58] Dipty. URL: http://www.dipity.com.

[59] IEEE Spectrum. URL: http://spectrum.ieee.org/.

[60] Adept MobileRobots. URL: http://www.mobilerobots.com/.

[61] I. Farkhatdinov, J.-H. Ryu, and J. Poduraev. "A feasibility study of time-domain passivity approach for bilateral teleoperation of mobile manipulator". In: *Control, Automation and Systems, 2008. ICCAS*

2008. International Conference on. 2008, pp. 272 –277. DOI: 10.1109/ICCAS.2008.4694563.

[62] Society of Robots. URL: http://www.societyofrobots.com/robot_arm_tutorial.shtml.

[63] P. J. McKerrow. *Introduction to Robotics*. Sydney: Addison-Wesley Pub. Co., 1991.

[64] C. Samson and M. Borgne. *Robot Control*. New York: Oxford University Press, 1990.

[65] Y. Nakamura. *Advanced Robotics: Redundancy and Optimization*. Reading, Mass.: Addison-Wesley Pub. Co., 1991.

[66] Carnegie Mellon. URL: http://www.cs.cmu.edu/.

[67] P. Kadleček. *A Practical Survey of Haptic APIs*. Tech. rep. Department of Software and Computer Science Education, 2010.

[68] D. Lee, O. Martinez-Palafox, and M. W. Spong. "Bilateral Teleoperation of a Wheeled Mobile Robot over Delayed Communication Network". In: *IEEE International Conference on Robotics and Automation*. Orlando, Florida, 2006, pp. 3298–3303.

[69] R. C. Goertz. "Fundamentals of General-Purpose Remote Manipulators". In: *Nucleonics* 10.11 (1952), pp. 36–45.

[70] R. C. Goertz. "Mechanical Master-Slave Manipulator". In: *Nucleonics* 12.11 (1954), pp. 45–46.

[71] W. Wei and Y. Kui. "Teleoperated manipulator for leak detection of sealed radioactive sources". In: *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*. Vol. 2. 2004, 1682 –1687 Vol.2. DOI: 10.1109/ROBOT.2004.1308066.

[72] R. Murphy. "Trial by fire [rescue robots]". In: *Robotics Automation Magazine, IEEE* 11.3 (2004), pp. 50 –61. ISSN: 1070-9932. DOI: 10.1109/MRA.2004.1337826.

[73] R. Uhrich. "Terminus controlled deep ocean manipulator". In: *Engineering in the Ocean Environment, Ocean 73 - IEEE International Conference on*. 1973, pp. 301 –304. DOI: 10.1109/OCEANS.1973.1161223.

[74] A. K. Bejczy. "Toward advanced teleoperation in space". In: *Teleoperation and Robotics in Space* (1994), p. 107. URL: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Toward+advanced+teleoperation+in+space#0.

[75] S. Lee, G. Bekey, and A. Bejczy. "Computer control of space-borne teleoperators with sensory feedback". In: *Robotics and Automation. Proceedings. 1985 IEEE International Conference on.* Vol. 2. 1985, pp. 205 –214. DOI: 10.1109/ROBOT.1985.1087390.

[76] L. Jenkins. "Telerobotic work system-space robotics application". In: *Robotics and Automation. Proceedings. 1986 IEEE International Conference on.* Vol. 3. 1986, pp. 804 –806. DOI: 10.1109/ROBOT.1986.1087553.

[77] A. Bejczy and Z. Szakaly. "Universal computer control systems (UCCS) for space telerobots". In: *Robotics and Automation. Proceedings. 1987 IEEE International Conference on.* Vol. 4. 1987, pp. 318 –324. DOI: 10.1109/ROBOT.1987.1087997.

[78] G. Hirzinger. "The space and telerobotic concepts of the DFVLR ROTEX". In: *Robotics and Automation. Proceedings. 1987 IEEE International Conference on.* Vol. 4. 1987, pp. 443 –449. DOI: 10.1109/ROBOT.1987.1088041.

[79] G. Hirzinger, J. Heindl, and K. Landzettel. "Predictive and knowledge-based telerobotic control concepts". In: *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on.* 1989, 1768 –1777 vol.3. DOI: 10.1109/ROBOT.1989.100231.

[80] G. Hirzinger et al. "Sensor-based space robotics-ROTEX and its telerobotic features". In: *Robotics and Automation, IEEE Transactions on* 9.5 (1993), pp. 649 –663. ISSN: 1042-296X. DOI: 10.1109/70.258056.

[81] T. Imaida et al. "Ground-space bilateral teleoperation of ETS-VII robot arm by direct bilateral coupling under 7-s time delay condition". In: *Robotics and Automation, IEEE Transactions on* 20.3 (2004), pp. 499 –511. ISSN: 1042-296X. DOI: 10.1109/TRA.2004.825271.

[82] W.-K. Yoon et al. "Model-based space robot teleoperation of ETS-VII manipulator". In: *Robotics and Automation, IEEE Transactions*

on 20.3 (2004), pp. 602 –612. ISSN: 1042-296X. DOI: 10.1109/TRA.2004.824700.

[83] G. H. Ballantyne. "Robotic surgery, telerobotic surgery, telepresence, and telementoring. Review of early clinical results". In: *Surg Endosc* 16.10 (2002), pp. 1389–1402.

[84] S.-G. Hong, J.-J. Lee, and S. Kim. "Generating artificial force for feedback control of teleoperated mobile robots". In: *Intelligent Robots and Systems, 1999. IROS '99. Proceedings. 1999 IEEE/RSJ International Conference on*. Vol. 3. 1999, 1721 –1726 vol.3. DOI: 10.1109/IROS.1999.811726.

[85] H. R. Otto J Rösch Klaus Schilling. "Haptic interfaces for the remote control of mobile robots". In: *Control Engineering Practice* 10.11 (2002), pp. 1309–1313.

[86] R. R. Rosenberg and D. C. Karnopp. *Introduction to Physical System Dynamics*. 1st. New York, NY, USA: McGraw-Hill, 1983.

[87] R. J. Anderson and M. W. Spong. "Hybrid Impedance Control of Robotic Manipulators". In: *IEEE Journal of Robotics and Automation* 4.5 (1988), pp. 549–556.

[88] B. Hannaford. "A design framework for teleoperators with kinesthetic feedback". In: *Robotics and Automation, IEEE Transactions on* 5.4 (1989), pp. 426 –434. ISSN: 1042-296X. DOI: 10.1109/70.88057.

[89] R. Chipman. *Transmission Lines*. McGraw-Hill, 1968.

[90] R. Anderson and M. Spong. "Asymptotic stability for force reflecting teleoperators with time delays". In: *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*. 1989b, 1618 – 1625 vol.3. DOI: 10.1109/ROBOT.1989.100209.

[91] A. de Rinaldis, R. Ortega, and M. W. Spong. "A compensator for attenuation of wave reflections in long cable actuator-plant interconnections with guaranteed stability". In: *Automatica* 42 (10 2006), pp. 1621–1635. ISSN: 0005-1098. DOI: http://dx.doi.org/10.1016/j.automatica.2006.05.011. URL: http://dx.doi.org/10.1016/j.automatica.2006.05.011.

[92] M. W. Spong and M. Vidyasagar. *Robot Dynamics and Control*. 1st. New York, NY, USA: John Wiley & Sons, Inc., 1989.

[93] T. Namerikawa and H. Kawada. "Symmetric Impedance Matched Teleoperation with Position Tracking". In: *Decision and Control, 2006 45th IEEE Conference on*. 2006, pp. 4496 –4501. DOI: 10.1109/CDC.2006.377743.

[94] R. Lozano, N. Chopra, and M. Spong. "Passivation Of Force Reflecting Bilateral Teleoperators With Time Varying Delay". In: *in Proceedings of the 8. Mechatronics Forum*. 2002, pp. 24–26.

[95] E. Nuño et al. "Position Tracking for Non-linear Teleoperators with Variable Time Delay". In: *The International Journal of Robotics Research* 28.7 (2009), pp. 895–910.

[96] E. Nuño, L. Basañez, and R. Ortega. "Passivity-based control for bilateral teleoperation: A tutorial". In: *Automatica* 47.3 (2011), pp. 485–495.

[97] M. Takegaki and S. Arimoto. "A New Feedback Method for Dynamic Control of Manipulators". In: *Journal of Dynamic Systems, Measurment, and Control of Manipulators* 103.2 (1981), pp. 119–125.

[98] D. Lee and M. W. Spong. "Passive Bilateral Teleoperation With Constant Time Delay". In: *IEEE Transactions on Robotics* 22.2 (2006), pp. 269–281.

[99] E. Nuño et al. "A Globally Stable PD Controller for Bilateral Teleoperators". In: *Robotics, IEEE Transactions on* 24.3 (2008), pp. 753 –758.

[100] E. Nuño et al. "On the asymptotic zero-convergence of position error for teleoperated robots with variable time-delay". In: *IEEE International Conference on Robotics and Automation*. 2008, pp. 2–9.

[101] N. Chopra and M. Spong. "Pasivity-Based Control of Multi-agent Systems". In: *Advances in Robot Control* (2006), pp. 107–134.

[102] N. Chopra and M. Spong. "Adaptive Synchronization of Bilateral Teleoperators with Time Delay". In: *Advances in Telerobotics* (2007), pp. 257–270.

[103] N. Chopra and M. Spong. "Delay-independent stability for interconnected nonlinear systems with finite L2 gain". In: *Decision and Control, 2007 46th IEEE Conference on*. 2007, pp. 3847 –3852. DOI: 10.1109/CDC.2007.4434672.

[104] W. Wang and J.-J. Slotine. "Contraction analysis of time-delayed communications and group cooperation". In: *Automatic Control, IEEE Transactions on* 51.4 (2006), pp. 712 –717. ISSN: 0018-9286. DOI: 10.1109/TAC.2006.872761.

[105] R. Ortega and M. Spong. "Adaptive motion control of rigid robots: a tutorial". In: *Decision and Control, 1988., Proceedings of the 27th IEEE Conference on*. 1988, 1575 –1584 vol.2. DOI: 10.1109/CDC.1988.194594.

[106] P. Miller, L.-F. Lee, and V. Krovi. "Output Synchronization for Teleoperation of Wheel Mobile Robot". In: *ASME Conference Proceedings* 2009.48937 (2009), pp. 707–714. DOI: 10.1115/DSCC2009-2637. URL: http://link.aip.org/link/abstract/ASMECP/v2009/i48937/p707/s1.

[107] N. Chopra and M. Spong. "Adaptive coordination control of bilateral teleoperators with time delay". In: *Decision and Control, 2004. CDC. 43rd IEEE Conference on*. Vol. 5. 2004, 4540 –4547 Vol.5. DOI: 10.1109/CDC.2004.1429499.

[108] S. Lee et al. "Haptic Control of a Mobile Robot: A User Study". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Lausanne, Switzerland, 2002.

[109] O.Khatib. "Real-time obstacle avoidance for manipulators and mobile robots". In: *International Journal of Robotics Research* 5.1 (1986).

[110] I. Farkhatdinov, J.-H. Ryu, and J. An. "A Preliminary Experimental Study on Haptic Teleoperation of Mobile Robot with Variable Force Feedback Gain". In: *Haptics Symposium, 2010 IEEE*. Waltham, Massachusetts, USA, 2010, pp. 251–256.

[111] Y. Chen et al. "Study on Coordinated Control and Hardware System of a Mobile Manipulator". In: *Intelligent Control and Automa-*

tion, 2006. WCICA 2006. The Sixth World Congress on. Vol. 2. 2006, pp. 9037 –9041. DOI: 10.1109/WCICA.2006.1713748.

[112] G. Niemeyer and J.-J. E. Slotine. "Telemanipulation with Time Delays". In: *International Journal of Robotics Research* 23.9 (2004), pp. 873–890.

[113] D. Lee and P. Y. Li. "Passive Bilateral Control and Tool Dynamics Rendering for Nonlinear Mechanical Teleoperators". In: *IEEE Transactions on Robotics* 21.5 (2005), pp. 936–951.

[114] J. Park and O. Khatib. "Robust Haptic Teleoperation of a Mobile Manipulation Platform." In: *ISER'04*. 2004, pp. 543–554.

[115] H. Tai and T. Murakami. "An approach to bilateral control strategy for nonidentical master-slave system". In: *Industrial Electronics, 2009. IECON '09. 35th Annual Conference of IEEE*. 2009, pp. 4173 –4178. DOI: 10.1109/IECON.2009.5415082.

[116] ROS tf. URL: http://ros.org/wiki/tf.

[117] J. Baillieul. "Kinematic programming alternatives for redundant manipulators". In: *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*. Vol. 2. 1985, pp. 722 –728. DOI: 10.1109/ROBOT.1985.1087234.

[118] P. Chang. "A closed-form solution for inverse kinematics of robot manipulators with redundancy". In: *Robotics and Automation, IEEE Journal of* 3.5 (1987), pp. 393 –403. ISSN: 0882-4967. DOI: 10.1109/JRA.1987.1087114.

[119] B. Siciliano and O. Khatib, eds. *Springer Handbook of Robotics*. Springer, 2008. ISBN: 978-3-540-23957-4.

[120] QUARC. URL: http://www.quarcservice.com/.

[121] A. Silva et al. "PHANToM OMNI Haptic Device: Kinematic and Manipulability". In: *Electronics, Robotics and Automotive Mechanics Conference, 2009. CERMA '09.* 2009, pp. 193 –198. DOI: 10.1109/CERMA.2009.55.

[122] Schunk GmbH & Co.KG. URL: http://www.schunk.com/.

[123] S. Bennett. "Nicholas Minorsky and the automatic steering of ships". In: *Control Systems Magazine, IEEE* 4.4 (1984), pp. 10 –15.

[124] H. Khalil. *Nonlinear Systems*. Prentice Hall, 2002.

[125] Gazebo. URL: http://gazebosim.org/.

[126] ROS robot_state_publisher. URL: http://www.ros.org/wiki/robot_state_publisher.

[127] Eigen/SVD. URL: http://eigen.tuxfamily.org/dox-2.0/classEigen_1_1SVD.html.

[128] Debian. URL: http://packages.debian.org/lenny/libraw1394-8.

[129] J. Wyatt et al. "Energy concepts in the state-space theory of nonlinear n-ports: Part I-Passivity". In: *Circuits and Systems, IEEE Transactions on* 28.1 (1981), pp. 48 –61. ISSN: 0098-4094. DOI: 10.1109/TCS.1981.1084907.

[130] H. K. Khalil. *Nonlinear Systems*. 3rd. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.

[131] M. W. Spong, S. Hutchinson, and M. Vidyasagar, eds. *Robot Modeling and Control*. John Wiley & Sons, Inc., 2006.

Appendix A

Detailed Background Theory

A.1 Passivity

A system is said to be passive if the system consumes energy, but does not produce energy. By definition [129], an n-port electrical system, with voltage $v(t)$ and current $i(t)$, is passive if the available energy E_A, defined as

$$E_A(x) = \sup_{x_0 \to T \geq 0} \int_0^T -\langle v(t), i(t) \rangle \, \mathrm{dt}, \tag{A.1}$$

is finite for all initial states x_0.

In addition, a system is passive if the energy consumed by the network in a time interval $[0, T]$ is greater than or equal to the increase in the energy stored in the network over the same period [130]. This is equivalent to

$$\int_0^T v(t) i(t) \, \mathrm{dt} \geq V(x(t)) - V(x_0), \tag{A.2}$$

where $V(x)$ is the energy stored in the network and x_0 the initial states.

Appendix B

Detailed Derivation of the Kinematics

B.1 Denavite-Hartenberg Representation

The forward kinematics can be found by the Denavite-Hartenberg (D-H) convention by using the following algorithm [131],

Step 1: Locate and label the joint axes z_0, ..., z_{n-1}.

Step 2: Establish the base frame. Set the origin anywhere on the z_0-axis. The x_0 and y_0 axes are chosen conveniently to form a right-hand frame.

For $i = 1, ..., n-1$, perform Steps 3 to 5.

Step 3: Locate the origin $\mathbf{o}_i$ where the common normal to z_i and z_{i-1} intersects z_i. If z_i intersects z_{i-1} locate $\mathbf{o}_i$ at this intersection. If z_i and z_{i-1} are parallel, locate $\mathbf{o}_i$ in any convenient position along z_i.

Step 4: Establish x_i along the common normal between z_{i-1} and z_i through $\mathbf{o}_i$, or in the direction normal to the $z_{i-1} - z_i$ plane if z_{i-1} and z_i intersect.

Step 5: Establish y_i to complete a right-hand frame.

Step 6: Establish the end-effector frame $\mathbf{o}_n x_n y_n z_n$. Assuming the n-th joint is revolute, set $z_n = a$ along the direction z_{n-1}. Establish the origin on conveniently along z_n, preferably at the center of the gripper or at the tip of any tool that the manipulator may be carrying. Set $y_n = s$ in the direction of the gripper closure and set $x_n = n$ as $s \times a$. If the tool is not a simple gripper set x_n and y_n conveniently to form a right-hand frame.

Step 7: Create a table of link parameters a_i, d_i, α_i, θ_i.

$a_i =$ distance along x_i from the intersection of the x_i and z_{i-1} axes to $\mathbf{o}_i$.

$d_i =$ distance along z_{i-1} from $\mathbf{o}_{i-1}$ to the intersection of the x_i and z_{i-1} axes. d_i is variable if joint i is prismatic.

$\alpha_i =$ the angle between z_{i-1} and z_i measured about x_i.

$\theta_i =$ the angle between x_{i-1} and x_i measured about z_{i-1}. θ_i is variable if joint i is revolute.

Step 8: Form the homogeneous transformation matrices A_i by substituting the above parameters into (B.1).

Step 9: Form $T_0^n = A_1 ... A_n$. This then gives the position and orientation of the tool frame expressed in base coordinates.

Each homogeneous transformation can be written as

$$A_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \qquad \text{(B.1)}$$

B.2 Singular Value Decomposition

The singular value decomposition (SVD) of an $m \times n$ real matrix A is a factorization of the form

$$A = U\Sigma V^T, \tag{B.2}$$

where U is an $m \times m$ unitary matrix, Σ an $m \times n$ rectangular diagonal matrix, and V^T an $n \times n$ unitary matrix. The diagonal elements of Σ, σ_i, are real, nonnegative and known as the singular values of A.

There are different approaches for solving the SVD, but it are closely related to the eigendecomposition. First, the non-zero singular values of A are the square roots of the non-zero eigenvalues of AA^T and A^TA, and chosen such that

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0,$$

where r are the number of non-zero singular values of A. Next, the columns of U, called the left singular vectors, and the columns of V, called the right singular values, corresponds to the eigenvectors of AA^T and A^TA, respectively.

The relationship between SVD and eigendecomposition can be seen by substituting A with $U\Sigma V^T$ in AA^T and A^TA

$$AA^T = U\Sigma V^T V\Sigma^T U^T = U\Sigma^2 U^T \Rightarrow (AA^T)U = U\Sigma^2, \tag{B.3}$$

and equivalent

$$A^TA = V\Sigma U^T U\Sigma^T V^T = V\Sigma^2 V^T \Rightarrow (A^TA)V = V\Sigma^2. \tag{B.4}$$

The matrix Σ can be partitioned in the following way

$$\Sigma = \begin{bmatrix} \Sigma_r & 0_{r\times(n-r)} \\ 0_{(m-r)\times r} & 0_{(m-r)\times(n-r)} \end{bmatrix}, \tag{B.5}$$

where Σ_r is an $r \times r$ diagonal matrix and r is the number of non-zero singular

values. Similarly

$$U = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} V_1 & V_2 \end{bmatrix}, \tag{B.6}$$

where U_1 and U_2 are the first r and last $m - r$ columns of U, respectively, and V_1 and V_2 are the first r and last $n-r$ columns of V, respectively. Which means that the matrix A can be written as $A = U_1 \Sigma_r V_1^T$.

B.3 Rotation Matrices for Phantom Omni

The rotating matrices for the Phantom Omni are calculated as

$$\begin{aligned}
R_1 &= \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad R_4 = \begin{bmatrix} \cos J_4 & 0 & -\sin J_4 \\ 0 & 1 & 0 \\ \sin J_4 & 0 & \cos J_4 \end{bmatrix} \\
R_2 &= \begin{bmatrix} \cos J_1 & 0 & \sin J_1 \\ 0 & 1 & 0 \\ -\sin J_1 & 1 & \cos J_1 \end{bmatrix} \quad R_5 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\cos J_5 & \sin J_5 \\ 0 & -\sin J_5 & -\cos J_5 \end{bmatrix} \\
R_3 &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(J_3 + J_{3,0}) & -\sin(J_3 + J_{3,0}) \\ 0 & \sin(J_3 + J_{3,0}) & \cos(J_3 + J_{3,0}) \end{bmatrix} \quad R_6 = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix},
\end{aligned} \tag{B.7}$$

where J_i are the joint angles of the joystick, shown in Figure 3.1 and Figure 3.4.

B.4 Homogeneous Transformation Matrices for Mobile Manipulator

By using the D-H convention in Section B.1 and the values in Table 3.1, the homogeneous matrices A_i can be written as

$$A_0 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & q_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & q_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} -\cos q_3 & -\sin q_3 & 0 & a_3 \cos q_3 \\ -\sin q_3 & \cos q_3 & 0 & a_3 \sin q_3 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_4 = \begin{bmatrix} \cos q_4 & 0 & \sin q_4 & 0 \\ \sin q_4 & 0 & -\cos q_4 & 0 \\ 0 & 1 & 0 & -d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_5 = \begin{bmatrix} \cos q_5 & 0 & -\sin q_5 & 0 \\ \sin q_5 & 0 & \cos q_5 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_6 = \begin{bmatrix} \cos q_6 & 0 & \sin q_6 & 0 \\ \sin q_6 & 0 & -\cos q_6 & 0 \\ 0 & 1 & 0 & -d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_7 = \begin{bmatrix} \cos q_7 & 0 & -\sin q_7 & 0 \\ \sin q_7 & 0 & \cos q_7 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_8 = \begin{bmatrix} \cos q_8 & 0 & \sin q_8 & 0 \\ \sin q_8 & 0 & -\cos q_8 & 0 \\ 0 & 1 & 0 & -d_8 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_9 = \begin{bmatrix} \cos q_9 & 0 & -\sin q_9 & 0 \\ \sin q_9 & 0 & \cos q_9 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_{10} = \begin{bmatrix} \cos q_{10} & -\sin q_{10} & 0 & 0 \\ \sin q_{10} & \cos q_{10} & 0 & 0 \\ 0 & 1 & 0 & -d_{10} \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where q_i are the generalized coordinates describing the configuration of the mobile manipulator.

Appendix C

Additional Implementation

C.1 Calculating the SVD

By using the Eigen/SVD-library [127] the SVD of a matrix $\mathbf{A}$ can be calculated in C++ programming language as

```
JacobiSVD<MatrixXf> svd(A, ComputeThinU | ComputeThinV);

double epsilon = std::numeric_limits
    <MatrixXf::Scalar>::epsilon();
MatrixXf::Scalar tolerance = epsilon*std::max(A.cols(),
    A.rows())*svd.singularValues().array().abs().maxCoeff();

MatrixXf result = svd.matrixV()*MatrixXf((svd.
    singularValues().array().abs() > tolerance).
    select(svd.singularValues().array().inverse(),0)).
    asDiagonal()*svd.matrixU().adjoint();
```

where `MatrixXf` is a matrix with dynamic size.