Model-predictive Collision Avoidance in Teleoperation of Mobile Robots

MODEL-PREDICTIVE COLLISION AVOIDANCE IN TELEOPERATION OF MOBILE ROBOTS

BY

SAJAD SALMANIPOUR, B.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2013) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Model-predictive Collision Avoidance in Teleoperation of Mobile Robots
AUTHOR:	Sajad Salmanipour B.Sc., (Electrical Engineering) Amirkabir University of Technology (Tehran Polytechnic), Tehran,Iran
SUPERVISOR:	Dr. Shahin Sirouspour
NUMBER OF PAGES:	xiii, 94

To my beloved parents

Bita & Habibollah

Abstract

In this thesis, a human-in-the-loop control system is presented to assist an operator in teleoperation of a mobile robot. In a conventional teleoperation paradigm, the human operator would directly navigate the robot without any assistance which may result in poor performance in complex and unknown task environments due to inadequacy of visual feedback. The proposed method in this thesis builds on an earlier general control framework that systematically combines teleoperation and autonomous control subtasks. In this approach, the operator controls the mobile robot (slave) using a force-feedback haptic interface (master). Teleoperation control commands coordinate master and slave robots while an autonomous control subtask helps the operator avoid collisions with obstacles in the robot task environment by providing corrective force feedback. The autonomous collision avoidance is based on a Model Predictive Control (MPC) philosophy. The autonomous subtask control commands are generated by formulating and solving a constrained optimization problem over a rolling horizon window of time into the future using system models to predict the operator force and robot motion. The goal of the optimization is to prevent collisions within the prediction horizon by applying corrective force feedback, while minimizing interference with the operator teleoperation actions. It is assumed that the obstacles are stationary and sonar sensors mounted on the mobile robot measure the obstacle

distances relative to the robot. Two formulation of MPC-based collision avoidance are proposed. The first formulation directly incorporates raw observation points as constraints in the MPC optimization problem. The second formulation relies on a line segment representation of the task environment. This thesis employs the well-known Hough transform method to effectively transform the raw sensor data into line segments. The extracted line segments constitute a compact model for the environment that is used in the formulation of collision constraints. The effectiveness of the proposed model-predictive control obstacle avoidance schemes is demonstrated in teleoperation experiments where the master robot is a 3DOF haptic interface and the slave is a P3-DX mobile robot equipped with eight (8) sonar sensors at the front.

Acknowledgements

Foremost, I would like to express my sincere appreciation to my supervisor, Dr. Shahin Sirouspour, for his patience, encouragement and support through my masters at McMaster University. I would like to also thank my committee members, Dr. Shahram Shirani and Dr. Thia Kirubarajan for their interest in my work.

I would like to record my warmest gratitude to my parents and brothers since their love and endless support have made it possible for me to excel in my studies.

My kind regards to my fellow colleagues in the Haptics, Telerobotics and Computational Vision Laboratory, who I have had the pleasure of working with. Special thanks to my friends and lab mates, Saman Rahnamaei and Behzad Iranpanah for their intellectual supports and suggestions.

Last but not the least, I am particularly grateful to my friends in Hamilton for all the fun and unforgettable moments we had together.

Notation and abbreviations

SMSS	Single-Master Single-Slave
SMMS	Single-Master Multiple-Slave
MMSS	Multiple-Master Single-Slave
MMMS	Multiple-Master Multiple-Slave
DOM	Degrees Of Mobility
DOF	Degrees Of Freedom
HMI	Human Machine Interface
TCF	Teleoperation Control Frame
MPC	Model Predictive Control
KDSR	Kinematically Deficient Slave Robot
KRSR	Kinematically Redundant Slave Robot

Contents

List of Figures

Chapter 1

Introduction

The prefix "tele" is a Greek originated word which signifies "at a distance", so teleoperation means operating at a distance (Hokayem and Spong, 2006). According to Sheridan (1989), *"teleoperation is the extension of a person's sensing and manipulation capability to a remote location"*. Early teleoperation applications emerged in space, nuclear and deep-water environments where complexity and uncertainty of the task environment required some level of human involvement, but having human present in the task environment was infeasible due to cost and safety issues (Ferre *et al.*, 2007). Since then, teleoperation systems have also been utilized to extend human capabilities and performance in tasks which are difficult to accomplish by the human operator directly due to scale and accessibility constraints. Examples include manipulation of very small or very large objects, Figure 1.1, and robotic-assisted surgery, Figure 1.2.

A teleoperation system generally consists of five basic elements as depicted in Figure 1.3. Human operator employs a bidirectional human-machine interface, also

Figure 1.1: Macro telemanipulation; passive hydraulic teleoperation system. The Center for Compact and Efficient Fluid Power, University of Minnesota.

Figure 1.2: Telesurgery; da Vinci® Si HD Surgical System, ©2012 Intuitive Surgical, Inc.

known as haptic interface, to control the remote manipulator and receive force feedback from the task environment.

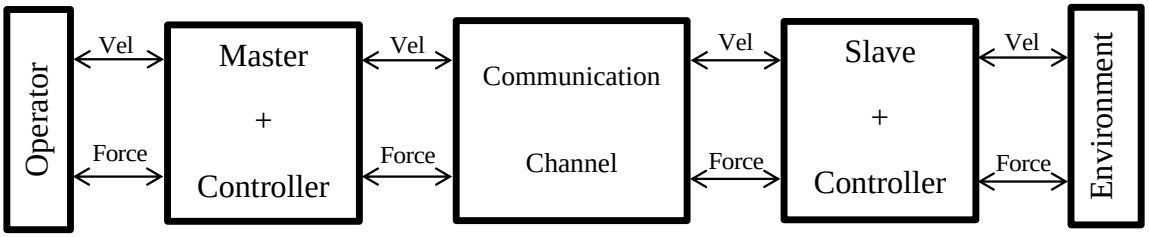

Figure 1.3: Basic teleoperation system elements.

A high performance teleoperation system guarantees velocity and position tracking between master and slave robots, and also provides the human operator with a sense of telepresence; as defined by Sheridan (1989), *"Telepresence is the ideal of sensing sufficient information about the teleoperator and task environment, and communicating this to the human operator in a sufficiently natural way, that the operator*

feels physically present at the remote site". Telepresence is measured in terms of transparency. A transparent haptic enabled teleoperation system allows the human operator to experience the same forces which have been applied to the slave robot at the remote task environment.

Teleoperation has been widely utilized in mobile robots applications (Diolaiti and Melchiorri, 2002; Lee *et al.*, 2005; Slawiñski *et al.*, 2012; Sanders, 2010; Lee *et al.*, 2006). Mobile robots are usually exploited in applications where a large work space must be covered. Search, rescue and surveillance operations (Nourbakhsh *et al.*, 2005), and space exploration (Schenker *et al.*, 2003) are examples of such applications where human cognitive capabilities combined with mobile robot extended workspace can compliment each other in order to accomplish the task objectives.

1.1 Motivation

A high performance teleoperation system should make the remote driving as easy as possible in remote operation of mobile robots. To reduce human operator cognitive work load and make telemanipulation straightforward, the idea of incorporating autonomous subtasks into teleoperation in an assistive way has been developed (Mallem *et al.*, 1992; Lumelsky and Cheung, 1993). Conventional teleoperation systems in which the human operator is responsible to control the slave robot without any assistance can result in poor performance in complex task environments. Consequently, more advanced teleoperation methods integrated with some level of autonomous control have been developed. Moreover, another important key factor which significantly affects cognitive workload is the means by which the Human Machine Interface (HMI) provides the human operator with all necessary information for carrying out the task

in the remote environment. In many telerobotic applications involving mobile robots, navigation is difficult due to the poor situation awareness which mainly arises from incomplete visual feedback information from the remote environment.

Tzafestas (2007) classifies different teleoperation paradigms and notes that computer-aided teleoperation schemes facilitate the task of human operator, see Figure 1.4. Advanced computer-aided teleoperation systems improve the teleoperation perfor-

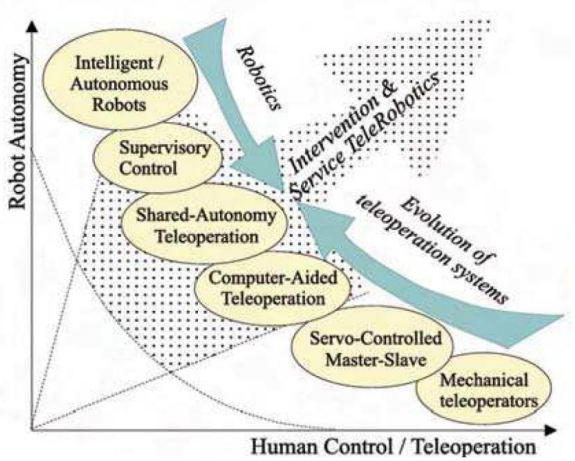

Figure 1.4: Evolution of teleoperation systems, adapted from Tzafestas (2007).

mance in one of the following ways (Tzafestas, 2007):

- Assisting the operator; that is by executing a set of autonomous operations, for example in the case of controlling some degrees of freedom in redundant manipulators, or in the case of satisfying safety related constraints.

- Improving perception; providing the human operator with enhanced sensory feedback, for example by utilizing graphical displays and haptic feedback.

In mobile robot teleoperation, collision-related concerns need not be handled directly by human operator. Employing an autonomous collision avoidance subtask

would assist the operator and make the remote navigation easier.

In mobile robot applications, operator is usually provided with visual feedback from the remote environment. Due to occlusion by obstacles and possible configurations in which the camera's field of view is restricted, relying merely on visual feedback could increase operator's cognitive workload and may result in judgment errors (Fong *et al.*, 2001). Lack of depth information in 2D cameras and complexity of systems providing 3D displays are other important drawbacks of vision feedback that necessitate specific trainings for the human operator to be effective in using such incomplete sensory feedback in navigation.

Alternative approaches have been proposed by Fong *et al.* (2001) and others to make navigation simpler and provide a better perception of the remote environment. Among those the use of haptic feedback to alert the operator to the presence of obstacles in the environment has shown significant reduction in operator's cognitive workload.

1.2 Problem Statement and Thesis Contributions

This thesis is concerned with developing a mixed autonomous/teleoperation control system for mobile robots where an autonomous subtask is employed to help the operator in navigation and collision avoidance. Designing such a system requires exploiting an obstacle avoidance method capable of being integrated into the mixed autonomous/teleoperation control system.

In recent years, many obstacle avoidance techniques have been developed for autonomous vehicles. Autonomous robots usually employ high-level trajectory planner, and methods for collision avoidance are based on this planned trajectory, local sensory

information and vehicle dynamics. Among all, approaches based on Model Predictive Control (MPC) have some advantages in comparison with other techniques (Liu *et al.*, 2010; Yoon *et al.*, 2009). MPC utilizes the system model, and by considering the planned trajectory and operational constraints (e.g. nearby obstacles, vehicle dynamics, etc.), obtains the optimal control actions over a window of time into the future to minimize a user-defined cost function.

However in teleoperation applications, a predefined path for the robot cannot be defined as the robot trajectory would depend on uncertain actions of the operator. The main contribution of this work is the development of an MPC-based obstacle avoidance algorithm and its integration within a general mixed autonomous/teleoperation control framework for telenavigation of mobile robots.

This thesis utilizes a general mixed autonomous/teleoperation control architecture developed by (Malysz, 2011; Malysz and Sirouspour, 2013). This framework is applicable to any system configuration, involving any number of operators, haptic devices, and robots. Among all different teleoperation configurations proposed by (Malysz, 2011; Malysz and Sirouspour, 2013), the Single-Master Single-Slave (SMSS) is of interest here where the human operator (single master) navigates the mobile robot (single slave). Along with teleoperation tasks, this framework allows for autonomous subtasks to be accomplished simultaneously. Based on a predefined priority level, the slave robot tracks both operator's desired commands and autonomous control commands.

As depicted in Figure 1.5, in the proposed approach of this thesis, the operator guides the mobile robot along the route, and collision avoidance subtask works in an assistive way to alert him/her about upcoming threats through force feedback. the operator

sends his/her desired commands through a haptic interface and also feels the movement of the mobile robot virtually; when there is no obstacles around, he/she only perceives mobile robot's mass, friction and physical constraints, e.g. nonholonomic motion constraints. When a collision is likely to happen in near future, the autonomous controlling command interferes with operator's desired commands to keep mobile robot away from obstacles; human operator senses this correction command and is informed about remote environment. The proposed control methodology is expected to reduce the operator's cognitive load and help him/her more easily navigate complex cluttered task environments than what would have been possible with merely visual feedback.

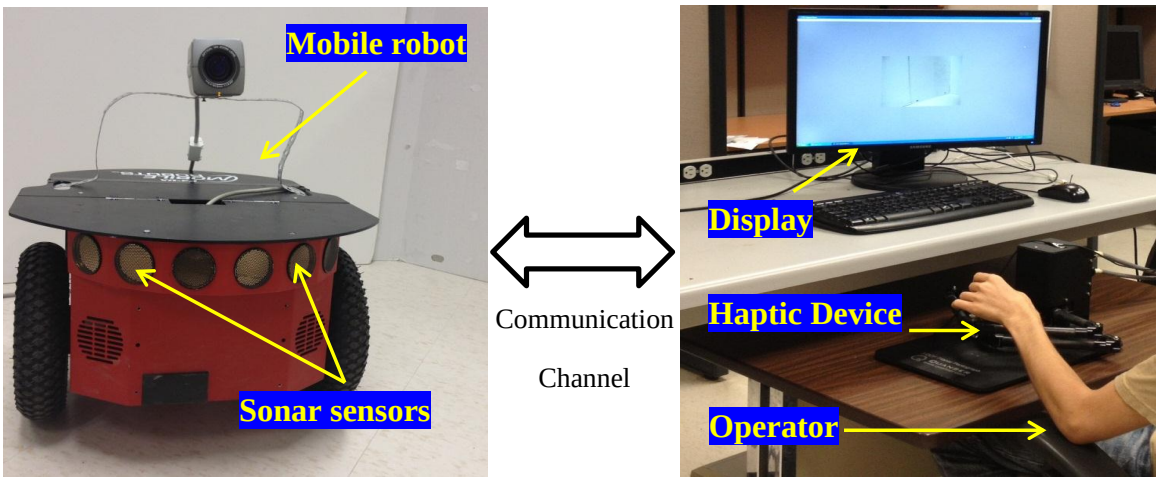

Figure 1.5: The human operator navigates the mobile robot and proper signals are sent to the haptic device to alert him/her about possible collisions.

To predict the system response over the control window, we need to derive a model for the overall teleoperation system and also predict the human operator input commands. To this end, a state-space model of the closed-loop system dynamics after the application of the controller in (Malysz, 2011) is developed. The states of

the system are assigned to be the mobile robot position, orientation and velocity. The system inputs consist of the operator force and a velocity-type command for the autonomous collision avoidance subtask. The operator force over the prediction window is estimated based on its past values. Collision avoidance is formulated as an autonomous subtask using the SMSS teleoperation configuration in (Malysz, 2011). This subtask runs at the same priority as that of the user-controlled teleoperation task. The subtask velocity commands are solutions to a constrained optimization problem formulated over a rolling horizon window of time into the future. The aim of this subtask is to prevent collisions with obstacles detected by the robot sonar sensors with minimum interference with the operator's teleoperation command.

Range-finder sonar sensors mounted on the mobile robot measure the obstacle distances relative to the robot. The outputs of the sensors can be essentially converted to points in a 2D Cartesian space. The obstacles are assumed to be stationary and a simple obstacle avoidance method is introduced first which incorporates observed raw points as constraints into the optimization problem. Since the optimization problem must be solved at each sample time, there would be a limit on the number of constraints and using raw points is not computationally efficient. Moreover, such a technique is sensitive to sonar measurement noise and various types of errors could occur (Iyengar and Elfes, 1991). This thesis addresses the basic problem of interpretation and utilization of these raw points in order to provide the optimization problem with an accurate and more compact form of the geometry of the remote environment. Considering obstacles and environment map to be a collection of features, many approaches have been recently developed to convert sensor raw data into geometric features, e.g. see (Madhavan *et al.*, 2002; Thrun *et al.*, 1998; Castellanos

et al., 1999). These methods often involve a trade-off between simplicity (requiring fewer computation) and accuracy (extracting as many features as possible). Among available geometric shapes, line segment is a simple model capable of describing many human made environments effectively and with an acceptable accuracy. This thesis employs the well-known Hough transform technique to effectively transform the raw sensor data into line segments and integrate the detected line segments as constraints into the MPC optimization problem formulation for collision avoidance.

1.3 Organization of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 provides a brief literature review on general teleoperation systems, teleoperation of mobile robots, interfaces used for collision avoidance, various types of obstacle avoidance methods and environment feature extraction using range-finder sensors. The general mixed autonomous/teleoperation framework proposed by (Malysz, 2011; Malysz and Sirouspour, 2013) is discussed in Chapter 3. In Chapter 4, the state-space model of the closed-loop system under the teleoperation controller is derived first, followed by a formulation of the MPC-based collision avoidance autonomous subtask control using original sensor measurements. Chapter 5 introduces a line segmentation method based on the Hough transform. It also presents a reformulation of the MPC obstacle avoidance algorithm so it can handle obstacles modeled by line segments. In Chapter 6, the system implementation and the results of experiments using a 3DOF haptic interface (as the master robot) and a P3-DX mobile robot (as slave) are presented. The thesis is concluded in Chapter 7 where some possible directions for future research are also discussed.

1.4 Related Publication

S. Salmanipour and S. Sirouspour, "Teleoperation of a Mobile Robot with Model-predictive Obstacle Avoidance Control", Accepted for presentation at the IECON 2013-39th Annual Conference on IEEE Industrial Electronics Society, IEEE, 2013.

Chapter 2

Literature Review

This Chapter is divided into four sections. The first section is a brief survey on conventional and advanced teleoperation systems. The second section reviews previous work on teleoperation of mobile robots and various types of HMI systems. The third section gives an overview of different types of collision avoidance methods. It also discusses advantages of MPC-based techniques in comparison with other methods of collision avoidance. The last section reviews some well-known algorithms for environment feature extraction and detecting obstacles.

2.1 Teleoperation

Early designs of teleoperation systems involved symmetric Single-Master Single-Slave teleoperation configurations. The term "symmetric SMSS" refers to the case when operator controls the slave robot via a master robot with the same Degree Of Mobility (DOM). Here a brief summary of SMSS teleoperation is given and thereafter, more complex architectures and recent developments on teleoperation systems are

discussed.

2.1.1 Symmetric SMSS Teleoperation

Early teleoperation systems were unilateral (Ferrell and Sheridan, 1967; Yoerger, 1982). In unilateral teleoperation, the controlling commands are one-directional signals from the master to the slave. Later on, Sheridan (1989) introduced the concept of *"telepresence"* which resulted in bilateral configuration for teleoperation systems.

Unlike the unilateral teleoperation which utilizes passive user interface, in bilateral teleoperation the human operator uses an active master device, also known as a haptic interface, to control a slave robot performing the task in the remote environment. The slave side measured position and/or force are fed back to the master side and the haptic interface generates mechanical signals to provide the operator with the sense of being present at the remote side, i.e. telepresence.

Two-port network models like the one in Figure 2.1 have long been used for modeling and analysis of teleoperation systems, e.g. see (Hashtrudi-Zaad and Salcudean, 2001; Sirouspour and Shahdi, 2005; Salcudean *et al.*, 2000; Arcara and Melchiorri, 2002).

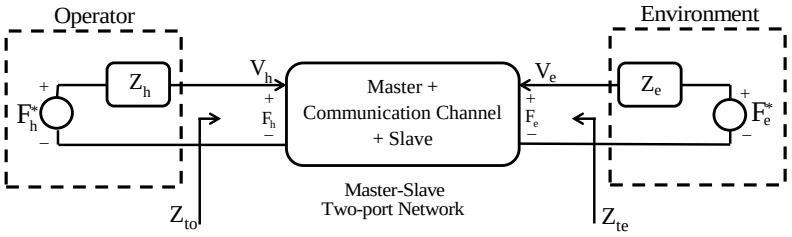

Figure 2.1: Two-port network model of a teleoperation system.

A hybrid matrix representation for two-port network is usually used for the analysis

of linear teleoperation systems, as:

$$\begin{bmatrix} F_h \\ -V_e \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} V_h \\ F_e \end{bmatrix} \tag{2.1}$$

Lawrence (1993) states that a transparent teleoperation system requires impedance matching and velocity tracking between master and salve sides, i.e.

$$Z_{to} = Z_e \tag{2.2}$$

$$V_h = V_e \tag{2.3}$$

where Z_{to} is defined as the transmitted impedance on the master side. Equation (2.2) is the impedance matching criterion that requires the impedance felt by human operator Z_{to} be equal to the environment impedance Z_e. Based on the hybrid matrix representation, one may calculate the master side transmitted impedance as

$$Z_{to} = \frac{h_{11} + (h_{11}h_{22} - h_{12}h_{21})Z_e}{1 + h_{22}Z_e} \tag{2.4}$$

Therefore, the hybrid matrix corresponding to an ideal transparent teleoperation system is given by

$$H_{ideal} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{2.5}$$

Implementation of such a transparent system requires four communication channels to provide master/slave side controllers with master/slave measured forces and

velocities (positions) (Yokokohji and Yoshikawa, 1994; Hashtrudi-Zaad and Salcudean, 2001).

The main challenges in controller design for the SMSS teleoperation are nonlinear, time-varying, and uncertain dynamics as well as communication delay. As a popular approach, the passivity-based controllers have been developed to guarantee the stability and transparency of the closed-loop teleoperation control system in the presence of communication delay and model uncertainties (Anderson and Spong, 1992; Lawrence, 1993; Yokokohji *et al.*, 2000; Niemeyer and Slotine, 2004).

2.1.2 Asymmetric Semi-autonomous Teleoperation

There are many other possible teleoperation configurations that do not fall under the symmetric SMSS category. Examples include:

- Systems involving multiple master and/or slave robots.
- Systems in which the master robot DOM is not same as the slave robot DOM.
- Configurations in which the human operator is not fully in charge of controlling the slave robot but instead is assisted by autonomous controllers to help with certain aspects of the task.

The first category covers three configurations, Multiple-Master Single-Slave (MMSS), Multiple-Master Multiple-Slave (MMMS) and Single-Master Multiple-Slave (SMMS) teleoperation systems. The MMSS architecture is generally used for cooperative control of slave robots (Katsura *et al.*, 2007). Sirouspour (2005) developed a general framework for MMMS teleoperation systems and studied a particular example in

which multiple slave robots hold a common tool to interact with the remote environment. The SMMS configuration is typically used in grasping (Lee and Spong, 2005) or when the operator wants to control the formation of multiple slave robots (Cheung *et al.*, 2009).

The second category includes control of Kinematically Redundant Slave Robots (KRSR) and Kinematically Deficient Slave Robots (KDSR). In the KRSR case, the extra slave DOM is controlled autonomously to help in handling singularities (Rubi *et al.*, 2002), avoiding possible collisions (Nanayakkara *et al.*, 2001) or any other desired subtasks. The KDSR teleoperation configuration is mostly used when the slave robot has physical constraints, such as in teleoperation of nonholonomic slave robots (Malysz and Sirouspour, 2011).

The third category relates to the integration of autonomous subtasks into teleoperation which has two basic applications: redundancy resolution to control the extra slave DOM (as mentioned before) and autonomous assistance in teleoperation of non-redundant slave robots. In the redundant case, since the operator does not control the nullspace, there is no interference between autonomous and teleoperation subtasks. For the non-redundant case, there is a mixed autonomous/teleoperation system. The autonomous subtask interferes with operator commands in an assistive way to guide the robot towards a desired area or to avoid forbidden regions. Abbott *et al.* (2007) proposed a virtual spring to assist the operator in avoiding possible singularities. Shared autonomous configuration also has been exploited to assist the operator in mobile robot navigation tasks (Diolaiti and Melchiorri, 2002; Poncela *et al.*, 2009).

The term "asymmetric semi-autonomous teleoperation" was first introduced by Malysz (2011) to cover all possible teleoperation configurations. Malysz and Sirouspour (2013) presented a unified teleoperation control framework which apart from being able to handle autonomous subtasks, allows for any number of master/slave robots with possible different DOM. This control framework yields a transparent and stable system and is the most general solution for teleoperation control problem to date.

2.2 Teleoperation of Mobile Robots

Mobile robots are increasingly used in applications where direct human presence in the task environment is infeasible or risky due to safety, distance, scale and other environmental barriers. Examples of such applications are in search and rescue and surveillance operations (Nourbakhsh *et al.*, 2005), maintenance of nuclear plants (Kim *et al.*, 2002b) and space exploration (Schenker *et al.*, 2003).

While significant progress has been made in autonomous operation of mobile robots, the complexity and uncertainty of many task environments necessitate some level of human involvement in the robot operation. In a teleoperation framework, the human operator would remotely control the mobile robot using passive or active (haptic) hand controllers. The operator awareness of the remote environment, often referred to as telepresence (Sheridan, 1989, 1992), is critical for the success of teleoperation. Video feeds are the most common form of feedback in teleoperation. However limited and/or obstructed camera field of view, lack of depth information in the 2D images, and the operator's preoccupation with other aspects of the task can potentially lead to collisions with obstacles (Alfano and Michel, 1990; Arthur, 2000).

To supplement visual feedback, (Nielsen *et al.*, 2007) presents an interface paradigm using a laser range-finder. However this approach could still increase operator's cognitive workload. In (Fong *et al.*, 2001), various types of man-machine interfaces are discussed for conveying information about potential collisions. Among those, a haptic device that allows the navigator to perceive obstacles through force feedback shows significant performance improvement.

2.3 Collision Avoidance Methods

Many approaches have been proposed for collision avoidance based on haptic feedback. A conventional way is to provide force feedback to the operator based on some measure of distance to the obstacle. For instance, obstacles may produce repulsive force fields to assist the operator navigating the scene (Diolaiti and Melchiorri, 2002; Lee *et al.*, 2006, 2005). Slawinski and Mut (Slawiñski *et al.*, 2012) found that augmenting obstacle related fictitious forces with predicted position of the vehicle one step ahead can reduce conservativeness of applying unnecessary forces.

Model predictive control (MPC) based obstacle avoidance strategies have been employed in autonomous robot control applications. A MPC-based obstacle avoidance algorithm for unmanned aerial vehicles was reported by Kim *et al.* (2002a). Other examples of MPC-based obstacle avoidance for autonomous vehicle systems can be found in Falcone *et al.* (2007), Keviczky *et al.* (2006) and Werling and Liccardo (2012). MPC-based strategies find optimal control actions by formulating and solving an optimization problem over a rolling horizon window into the future. This can potentially enhance the system performance by considering its future behavior

and offers a great degree of flexibility in defining various collision objectives and constraints. In autonomous vehicles, a desired path is predefined given a prior knowledge of the robot task environment. Then, the MPC guides the vehicle to follow a feasible obstacle-free modified version of the predefined path, considering robot limitations and real-time local sensor observations (Park *et al.*, 2009; Liu *et al.*, 2010). In a similar manner, Droge and Egerstedt (Droge and Egerstedt, 2011) introduced a receding horizon method which predicts future system states and chooses the optimal solution among some predefined specified actions. The work done by Howard *et al.* (2010) and Krüsi *et al.* (2010) addressed balancing issues arising from following the desired predefined path and satisfying local criteria (e.g obstacle avoidance). This specific trade-off problem is solved through defining a suitable cost function for MPC based methods or by employing other techniques such as deformable virtual zone (DVZ) (Lapierre *et al.*, 2007).

2.4 Environment Feature Extraction Using Range-finder Sensors

In real time implementation of obstacle avoidance algorithms, a huge amount of obstacles related scanned points measured by range-finder sensors like laser scanners or sonar sensors should be processed at each sample time; so transforming raw data into a more compact form is desirable. A popular way of achieving this objective is to extract geometric features of the surrounding environment from raw sensed data (Gutmann and Schlegel, 1996; Jensfelt and Christensen, 1998; Borges and Aldon, 2004; Nguyen *et al.*, 2007). These methods model the task environment in a compact

form in terms of elementary geometric features.

One of the widely used geometric features in environment modeling is line segment. A map constructed from line segments only is a middle ground which is not, neither a highly summarized complex map nor a massively redundant raw points map. Line-based maps are the most suitable choice to describe structured environments comprised of straight edged objects (Pfister *et al.*, 2003; Nguyen *et al.*, 2007). Premebida and Nunes (2005) discuss some methods for data segmentation and feature extraction from 2D range scans provided by a Laser Range Finder (LRF). Pfister *et al.* (2003) uses a weighted line fitting method to create a line-based map. Vandorpe *et al.* (1996) suggests a dynamic map building consisting of two features, lines and circles. Nguyen *et al.* (2007) presents an excellent survey, comparing different line extraction algorithms using 2D range data for indoor mobile robot applications. The line segments can be further processed as in the work by Zhang and Ghosh (2000), where they are used as basic elements to generate a closed and connected region.

The Hough transform is arguably one of the most successful line extraction algorithms applied on intensity images (Ponce and Forsyth, 2002). In 1962, Hough developed a means to represent Cartesian domain complex arrangements in a parameterized form (Hough, 1962). By exploiting this concept, Duda and Hart (1972) proposed a method for using Hough transform to detect lines in pictures which was extensively used not only in mobile robot research areas (Glennon, 1998; Jensfelt and Christensen, 1998; Pfister *et al.*, 2003) but also by those working in computer vision and image processing fields (Ballard, 1981; Horn, 1986). A few modifications have also been made to the original Hough transform to make it more efficient. For example, range findings at certain values are weighted more heavily and the *"Range*

Weighted Hough Transform" (RWHT) is used instead (Forsberg *et al.*, 1993, 1995; Larsson *et al.*, 1996).

Chapter 3

Mixed Autonomous/Teleoperation Control Framework

3.1 General Formulation

In this work, the general mixed autonomous/teleoperation control architecture by Malysz (2011) and Malysz and Sirouspour (2013) is used. This framework can handle any number of master and slave robots, symmetric and asymmetric scenarios, as well as different types of autonomous control commands. The focus of this thesis is on a SMSS configuration. In this case, two teleoperation control frames (TCF) are assigned to the master (X_m) and slave (X_s) robot end effectors and are coordinated via control.

A general overview of the control architecture by Malysz (2011) is given in Figure 3.1. As shown, the proposed control strategy is comprised of four high-level teleoperation-specific subtasks on top of low-level joint velocity controllers. The *"prioritized autonomous control"* subtasks have the highest priority level and the rest of

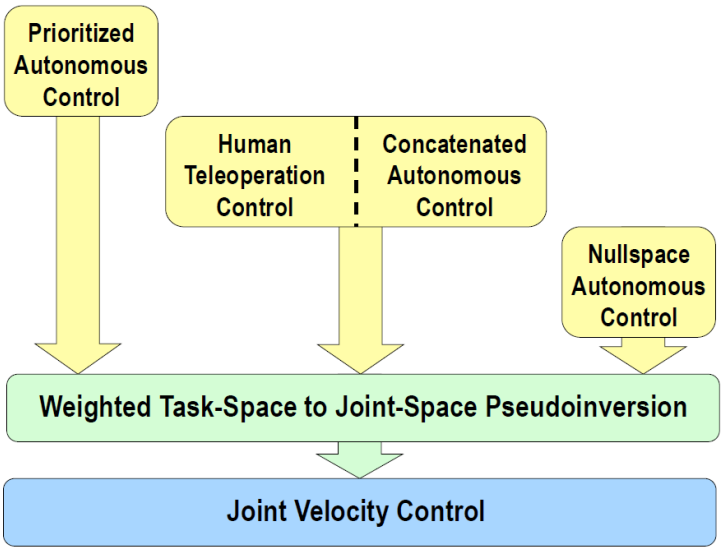

Figure 3.1: Conceptual overview of the mixed autonomous/teleoperation control strategy proposed by Malysz and Sirouspour (2013). The yellow blocks serve as high-level subtasks and their height represent their priority level. Adapted from Malysz and Sirouspour (2013).

the control subtasks should be defined in its null-space. The *"human teleoperation control"* relates to the master/slave tracking subtasks; it should be noted that the master device controls all or a subset of the DOF of the slave side TCF. Another set of subtasks are *"concatenated autonomous control"* which act on the same subset of DOF which the teleoperation related subtasks do. However, the relative strength between those can be adjusted via a weighting matrix. The lowest priority is given to the *"null-space autonomous control"* which is in charge of controlling any remaining null-space velocities.

As mentioned earlier, mobile robot navigation using teleoperation would benefit from human operator cognitive capabilities and adding assistive autonomous subtasks would decrease the operator workload. The *prioritized autonomous control* subtask would not be suitable for collision since it would completely override the operator's

teleoperation commands. Also since the two-wheeled nonholonomic mobile robots considered in this thesis fall under the KDSR category, they are not amenable to *null-space autonomous control*. As shown in the Figure 3.2, a two-wheeled mobile robot has 2DOF: translational velocity (ν) and rotational velocity (ω). The nonholonomic motion constraint is:

$$\dot{x}\sin\theta - \dot{y}\cos\theta = 0 \tag{3.1}$$

where $\dot{x}, \dot{y}$ are mobile robot velocities in the Cartesian space and θ is its orientation.

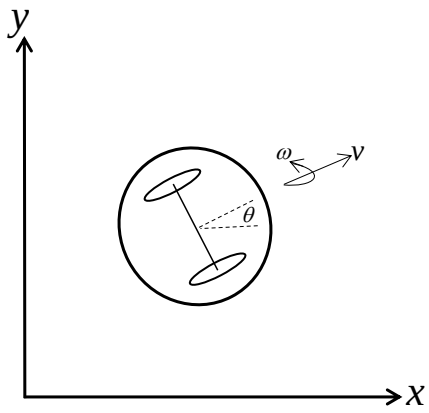

Figure 3.2: Nonholonomic two-wheeled mobile robot.

In this thesis the *human teleoperation control* and *concatenated autonomous control* for collision avoidance are considered. While according to the Figure 3.1, these tasks are defined at same priority level, their relative priority could still be adjusted through a user-selectable weighting matrix in the controller.

3.1.1 System Dynamics and Control Laws

The master and slave robot dynamics have the following nonlinear form:

$$M_m(q_m)\ddot{q}_m + C_m(q_m, \dot{q}_m)\dot{q}_m + D_m\dot{q}_m + g_m(q_m) = \tau_m + J_m^T(q_m)f_{e,m} \tag{3.2}$$

$$M_s(q_s)\ddot{q}_s + C_s(q_s, \dot{q}_s)\dot{q}_s + D_s\dot{q}_s + g_s(q_s) = \tau_s + J_s^T(q_s)f_{e,s} \tag{3.3}$$

$$M_{m/s}, C_{m/s}, D_{m/s} \in \mathbb{R}^{n_{m/s}\times n_x} \quad , \quad q_{m/s}, g_{m/s}, \tau_{m/s}, \in \mathbb{R}^{n_{m/s}}$$

$$J_{m/s} \in \mathbb{R}^{n_x\times n_{m/s}} \quad , \quad f_{e,m/s} \in \mathbb{R}^{n_x}$$

where $q_{m/s}$ is the master/slave joint variables vector, $M_{m/s}$ and $D_{m/s}$ represent mass and damping positive definite matrices, $C_{m/s}$ stands for centrifugal and Coriolis effects, $g_{m/s}$ relates to the position-dependent forces (e.g. gravity), $\tau_{m/s}$ is the vector of joint torques, and $J_{m/s}$ is the master/slave robot Jacobian matrix mapping workspace forces to joint space torques. $n_{m/s}$ is the DOM of the master/slave robot and n_x is the DOF of the Cartesian frame assigned to the master/slave robot.

The high-level teleoperation control block generates proper master/slave desired task space velocity commands by utilizing master/slave side measured forces, positions and velocities; the resulting commands are fed into a low-level joint space velocity control block after being multiplied by the generalized weighted pseudoinverse matrix which will be defined later on.

The velocity kinematics of the master and slave TCFs are governed by:

$$\dot{X}_m = J_m\dot{q}_m \qquad X_m \in \mathbb{R}^{n_x\times 1}, J_m \in \mathbb{R}^{n_x\times n_m}, q_m \in \mathbb{R}^{n_m\times 1} \tag{3.4}$$

$$\dot{X}_s = J_s\dot{q}_s \qquad X_s \in \mathbb{R}^{n_x\times 1}, J_s \in \mathbb{R}^{n_x\times n_s}, q_s \in \mathbb{R}^{n_s\times 1} \tag{3.5}$$

And the autonomous subtask is defined by following velocity-like signal:

$$\dot{c} = J_c \dot{q}_s \qquad\qquad J_c \in \mathbb{R}^{n_c \times n_s}, c \in \mathbb{R}^{n_c \times 1} \tag{3.6}$$

where n_c is the DOF of the selected frame for autonomous subtask. The velocity vectors $\dot{X}_s$ and $\dot{c}$ are concatenated to allow mixed autonomous/human teleoperation:

$$\dot{\bar{X}}_{sc} \triangleq \begin{bmatrix} \dot{X}_s \\ \dot{c} \end{bmatrix} = \begin{bmatrix} J_s \\ J_c \end{bmatrix} \dot{q}_s = J_{sc} \dot{q}_s \tag{3.7}$$

$$\bar{X}_{sc} \in \mathbb{R}^{(n_x+n_c)\times 1}, \quad J_{sc} \in \mathbb{R}^{(n_x+n_c)\times n_s}$$

One of teleoperation performance objectives is motion tracking between the master and slave TCFs. The corresponding tracking errors are defined as

$$\rho_e \triangleq X_s - k_p X_m \qquad\qquad k_p \in \mathbb{R}^{n_x \times n_x} \tag{3.8}$$

$$\dot{\rho}_e \triangleq \dot{X}_s - k_p \dot{X}_m \tag{3.9}$$

where k_p is a scaling matrix. The autonomous subtask control performance is characterized by the following tracking error

$$\rho_c \triangleq J_c(\dot{q}_s^d - \dot{q}_s) \tag{3.10}$$

where $\dot{q}_s^d$ is a desired joint-space velocity command that will be defined later. The teleoperation framework assumes that low-level joint velocity controllers are available

to ensure velocity tracking at the joint level, i.e.

$$\rho_m \triangleq \dot{q}_m^d - \dot{q}_m \in L_2 \cap L_\infty \tag{3.11}$$

$$\rho_s \triangleq \dot{q}_s^d - \dot{q}_s \in L_2 \cap L_\infty \tag{3.12}$$

An example of such controllers is the adaptive nonlinear controllers in Malysz (2011).

The high-level teleoperation commands are defined based on velocity commands in the workspace coordinates, which are given by

$$\begin{aligned}\dot{X}_m^d \triangleq\ & k_p^{-1} P_{uxx}[a(k_h \tilde{f}_m + k_f \tilde{f}_s) + \Lambda(X_s - k_p X_m) + (I-\Omega)(\Sigma \tilde{\dot{X}}_s + (I-\Sigma)k_p \tilde{\dot{X}}_m)] \\ & + K_p^{-1} P_{nxx}[(I-2\Sigma)(I-\Omega)(k_p \tilde{\dot{X}}_m - \tilde{\dot{X}}_s) + 2\Lambda(X_s - k_p X_m))] \\ & + k_p^{-1} P_{uxc} \dot{c}^d \end{aligned} \tag{3.13}$$

$$\dot{X}_s^d \triangleq\ a(k_h \tilde{f}_m + k_f \tilde{f}_s) + \Lambda(k_p X_m - X_s) + (I-\Omega)((I-\Sigma)\tilde{\dot{X}}_s + \Sigma k_p \tilde{\dot{X}}_m) \tag{3.14}$$

where $f_m, f_s \in \mathbb{R}^{n_x \times 1}$ are master and slave devices end-effector force measurements and a, Λ, Σ, Ω are positive diagonal matrices. The tilde subscript indicates filtered data according to the filter defined by Equation (3.15) in frequency domain, which is used to filter noisy measurements of velocities and forces.

$$\tilde{X}(s) = \frac{C}{s+C} X(s) \tag{3.15}$$

The projection matrices P_{uxx}, P_{nxx}, P_{uxc} are defined as

$$J_{sc}\bar{J}_{sc}^{\dagger} \triangleq P_u = \begin{bmatrix} P_{uxx} & P_{uxc} \\ P_{ucx} & P_{ucc} \end{bmatrix} \tag{3.16}$$

$$P_n = \begin{bmatrix} I - P_{uxx} & -P_{uxc} \\ -P_{ucx} & I - P_{ucc} \end{bmatrix} = \begin{bmatrix} P_{nxx} & P_{nxc} \\ P_{ncx} & P_{ncc} \end{bmatrix} \tag{3.17}$$

$$P_n, P_u \in \mathbb{R}^{(n_x+n_c)\times(n_x+n_c)}$$

$\bar{J}_{sc}^{\dagger}$ is the generalized pseudoinverse defined as

$$\bar{J}_{sc}^{\dagger} \triangleq [\bar{J}_{sc}^T W_{xc}^2 \bar{J}_{sc}]^{-1} \bar{J}_{sc}^T W_{xc}^2 \tag{3.18}$$

$$\bar{J}_{sc}^{\dagger} \in \mathbb{R}^{n_s\times(n_x+n_c)} \ , \ W_{xc} \in \mathbb{R}^{(n_x+n_c)\times(n_x+n_c)}$$

W_{xc} is the task-pace weighting matrix which would determine the relative dominance of the teleoperation and concatenated autonomous control subtasks, which has the following form

$$W_{xc} \triangleq \ diag(\sqrt{w_x} I_{nx}, \sqrt{w_c} I_{nc}) \tag{3.19}$$

$$w_x > 0 \ , \ w_c > 0$$

After designing the high-level task-space desired velocities and a generalized pseudoinverse for the Jacobian matrix to map from task-space to joint-space, the desired

joint-space velocities are chosen as

$$\dot{q}_m^d = J_m^{-1}\dot{X}_m^d \tag{3.20}$$

$$\dot{q}_s^d = \bar{J}_{sc}^{\dagger}\dot{\bar{X}}_{sc}^d = \bar{J}_{sc}^{\dagger}\begin{bmatrix}\dot{X}_s^d \\ \dot{c}^d\end{bmatrix} \tag{3.21}$$

where $\dot{c}^d$ is the autonomous subtask desired velocity. This terms will be determined by formulating and solving an optimization problem aimed at avoiding collisions with obstacles in the robot workspace.

Figure 3.3 provides a block diagram representation of the general teleoperation framework with a concatenated autonomous control subtask, Malysz (2011). The

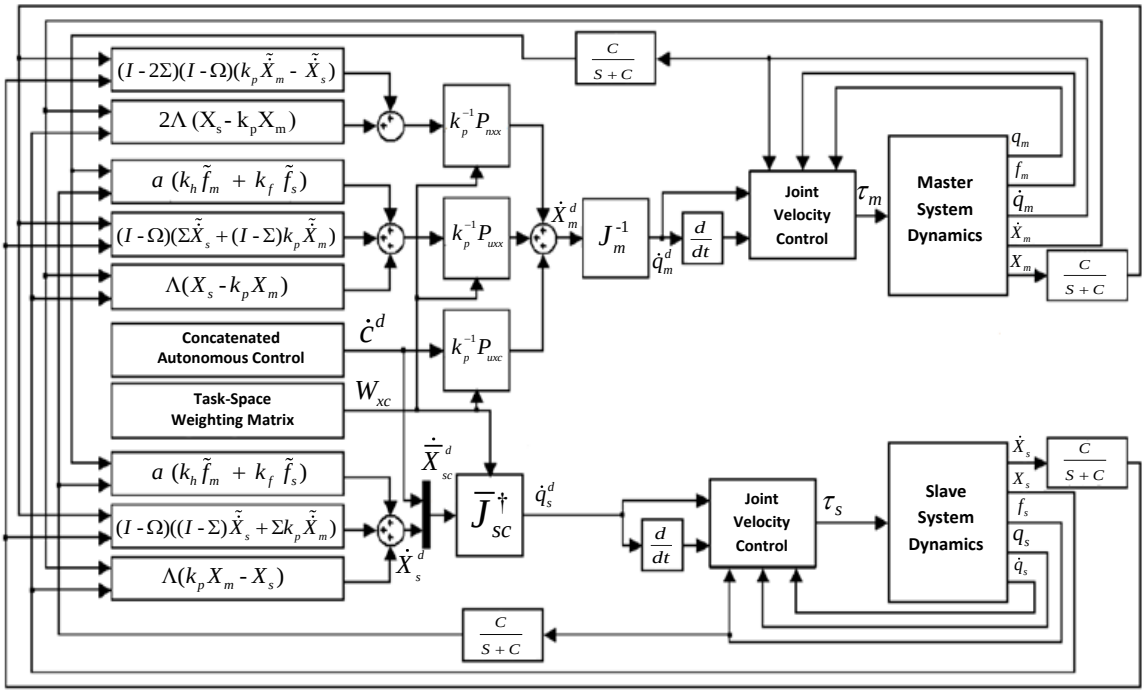

Figure 3.3: Block diagram of the general teleoperation control framework for the simplified case of including only concatenated autonomous control subtasks; adapted from Malysz (2011).

high-level teleoperation controller generates task-space velocity commands $\dot{X}_m^d, \dot{X}_s^d$ first and then the autonomous control command $\dot{c}^d$ is concatenated to the $\dot{X}_s^d$ in the form of $\dot{\bar{X}}_{sc}^d$ which allows for shared autonomous/human teleoperation control. These high-level commands are multiplied by generalized pseudoinversion Jacobian matrices to provide the low-level joint velocity controllers with desired joint-space velocity commands $\dot{q}_m^d, \dot{q}_s^d$.

Assuming (3.11) and (3.12), and by employing (3.13), (3.14), (3.20) and (3.21), Malysz (2011) proves that:

$$\rho_e,\ \dot{\rho}_e,\ \rho_c \in L_2 \cap L_\infty \tag{3.22}$$

which means $\rho_e,\ \dot{\rho}_e,\ \rho_c$ converges to zero; i.e. teleoperation and autonomous subtask motion tracking is achieved. It also can be shown that the operator would perceive a mechanical impedance and a constraint force due to the obstacle avoidance subtask, f_c, of the form

$$k_h f_m + k_f f_s + f_c = \underbrace{(I + P_{uxx}^{-1} P_{nxx}) a^{-1} C^{-1} k_p}_{\text{virtual mass}} \ddot{X}_s + \underbrace{(\Omega + P_{uxx}^{-1} P_{nxx}) a^{-1} k_p}_{\text{virtual damping}} \dot{X}_s \tag{3.23}$$

$$f_c \triangleq a^{-1} P_{uxx}^{-1} P_{uxc} \dot{c}^d \tag{3.24}$$

The f_c term represents virtual forces arising from the autonomous control subtask and C is the cutoff frequency of the filter defined by Equation (3.15). According to this equation, the teleoperation and autonomous control subtasks are in a mixed form resulting in shared semi-autonomous teleoperation control. The collision avoidance autonomous subtask velocity command $\dot{c}^d$ is the solution to an optimization problem

related to the obstacle avoidance algorithm, which will be discussed later in detail.

The wighting matrix affects the projection matrices. Choosing the concatenated autonomous control weight to be very small, i.e. $w_c \ll w_x$, results in $P_{uxx} \approx I, P_{nxx} \approx 0, P_{uxc} \approx 0$ which means the human operator would not feel virtual forces caused by autonomous control subtask. Increasing w_c which means giving more weight to the concatenated autonomous control subtask, decreases the minimum eigenvalue in teleoperation related projection matrix P_{uxx}; as a result the virtual force vector generated by autonomous subtask increases.

3.2 Mobile Robot Teleoperation

The focus of this work is teleoperation of a planar mobile with a nonholonomic motion constraint. The particular robot used in the experiments is a Mobile Robot P3-DX with two DOM ($n_s = 2$), shown in Figure 3.4. A 3DOF planar haptic interface from Quanser (see Figure 3.5) with four actuators ($n_m = 4$) is used to teleoperate the mobile robot. All the equations derived in the previous section are applicable to establish the semi-autonomous teleoperation control between the master and slave robots, except the f_s term; since the slave side mobile robot does not have force sensor, this term is set to be zero.

As shown in Figures 3.4 and 3.5, the position and orientation of the pantograph (X_m) and mobile robot (X_s) are chosen as the master and slave TCF coordinate variables with $n_x = 3$. The autonomous subtask TCF (c) is the same as the slave

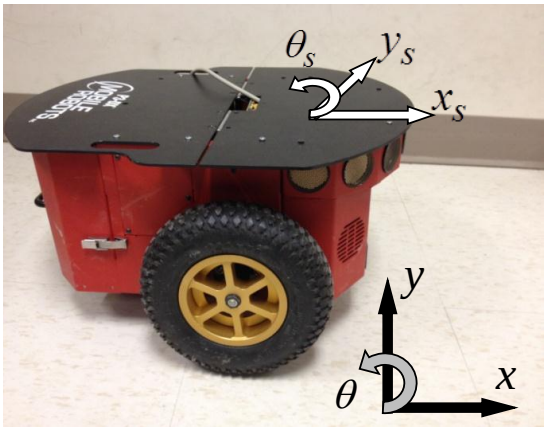

Figure 3.4: P3-DX mobile robot as the slave robot. Selected task space coordinate frame X_s is demonstrated. The world frame is depicted at the right bottom corner.

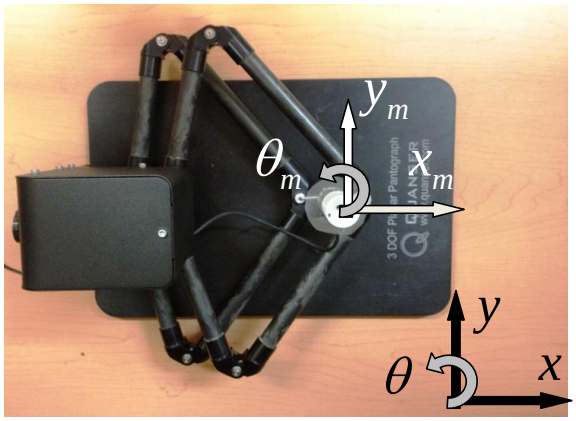

Figure 3.5: Pantograph as the master device. Selected task space coordinate frame X_m is demonstrated. The world frame is depicted at the right bottom corner.

side TCF ($n_c = 3$). The slave side velocity kinematics are given by

$$\dot{\bar{X}}_{sc} = \begin{bmatrix} \dot{X}_s \\ \dot{c} \end{bmatrix} = \begin{bmatrix} \dot{x}_s \\ \dot{y}_s \\ \dot{\theta}_s \\ \dot{c}_x \\ \dot{c}_y \\ \dot{c}_\theta \end{bmatrix} = \begin{bmatrix} J_s \\ J_c \end{bmatrix} \dot{q}_s = \bar{J}_{sc} \begin{bmatrix} \upsilon_s \\ \omega_s \end{bmatrix} \tag{3.25}$$

$$J_s = J_c = \begin{bmatrix} \cos\theta_s & 0 \\ \sin\theta_s & 0 \\ 0 & 1 \end{bmatrix} \tag{3.26}$$

where υ_s and ω_s are mobile robot translational and rotational velocities respectively. The master robot velocity kinematics are governed by:

$$\dot{X}_m = J_m \dot{q}_m = J_m \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} \tag{3.27}$$

$$J_m = \begin{bmatrix} \frac{-l_1 l_3 \sin q_1}{l_2} & \frac{l_1 l_3 \sin q_2}{l_2} & \frac{l_1 (l_2+l_3) \sin q_3}{l_2} & \frac{-l_1 (l_2+l_3) \sin q_4}{l_2} \\ \frac{-l_1 l_3 \cos q_1}{l_2} & \frac{-l_1 l_3 \cos q_2}{l_2} & \frac{-l_1 (l_2-l_3) \cos q_3}{l_2} & \frac{-l_1 (l_2-l_3) \cos q_4}{l_2} \\ \frac{l_1 \sin q_1}{l_2 \cos \theta_m} & \frac{-l_1 \sin q_2}{l_2 \cos \theta_m} & \frac{-l_1 \sin q_3}{l_2 \cos \theta_m} & \frac{-l_1 \sin q_4}{l_2 \cos \theta_m} \end{bmatrix} \tag{3.28}$$

The pantograph parameters are depicted in the Figure 3.6.

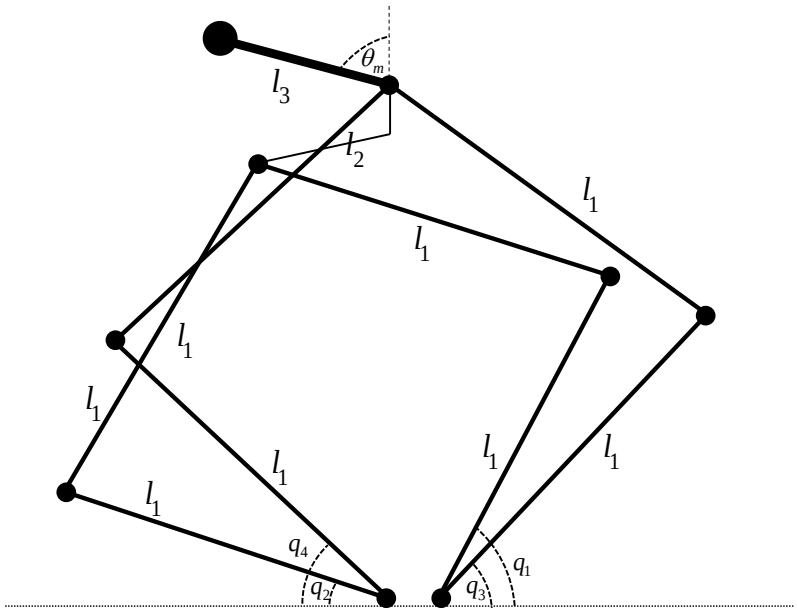

Figure 3.6: Master side planar robot; schematic and parameters.

Chapter 4

Model Predictive Collision Avoidance Algorithm

In this chapter, the mixed autonomous/teleoperation system proposed by Malysz (2011) is modeled and the MPC-based collision avoidance subtask control is formulated. The obstacles are assumed to be stationary and original sensor measurements are used to avoid local obstacles.

4.1 System Model

A state-space model of the closed-loop system dynamics after the application of the teleoperation controller is needed for the design of MPC-based collision avoidance autonomous subtask command $\dot{c}^d$. Equations (3.8), (3.9), (3.22) guarantee motion tracking between master and slave side, so the slave side position and velocity can be

approximated as

$$X_s \simeq K_p X_m \tag{4.1}$$

$$\dot{X}_s \simeq K_p \dot{X}_m \tag{4.2}$$

The slave TCF coordinates (X_s) and their derivatives ($\dot{X}_s$) are selected as the states and, f_m and $\dot{c}^d$ as the inputs of the system. With these choices, the state space representation of the mixed autonomous/teleoperation system is given by

$$\begin{aligned}
&\dot{\mathcal{X}} = A(\theta_s)\mathcal{X} + B(\theta_s)U \qquad (4.3)\\
&\mathcal{X} = \begin{bmatrix} X_s \\ \dot{X}_s \end{bmatrix} \ , \ U = \begin{bmatrix} f_m \\ \dot{c}^d \end{bmatrix}\\
&f_m = \begin{bmatrix} f_x \\ f_y \\ \tau \end{bmatrix} \ , \ \dot{c}^d = \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix}\\
&\mathcal{X}, U \in \mathbb{R}^{2n_x \times 1}
\end{aligned}$$

The A and B matrices are obtained using (3.23) as

$$\begin{aligned}
A(\theta_s) &= \begin{bmatrix} 0 & I \\ 0 & C(P_{uxx}(I-\Omega)-I) \end{bmatrix} \qquad (4.4)\\
B(\theta_s) &= \begin{bmatrix} 0 & 0 \\ CP_{uxx}a & CP_{uxc} \end{bmatrix}
\end{aligned}$$

Using (3.16), (3.17), (3.18), (3.26) and also choosing $W_{xc} = I$, the projection matrices are calculated as

$$P_{uxx} = P_{uxc} = \begin{bmatrix} \cos^2\theta_s & \cos\theta_s \sin\theta_s & 0 \\ \cos\theta_s \sin\theta_s & \sin^2\theta_s & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.5}$$

It should be noted that the dynamics in (4.3) are nonlinear due to dependency of A and B on the mobile robot orientation θ_s.

4.2 Model Predictive Control for Collision Avoidance

In this section an optimization problem over a rolling horizon window is formulated to find the collision avoidance subtask velocity command $\dot{c}^d$. First, an overview of the optimization problem is given. Next, the constraints and cost function will be discussed in details.

4.2.1 Problem Definition

To find a solution for the $\dot{c}^d$ in order to assist the human operator to avoid collisions, the mobile robot planned path and locations of obstacles should be determined. 2D Range finder sensors (e.g. sonar sensors) could be used to detect local obstacles. Since the slave mobile tracks the human operator desired trajectory, the planned path is not known a priori.

The discrete-time state-space representation of the closed-loop teleoperation system

can be used for predicting the future mobile robot states.

The system model in (4.3) is in the continuous time domain and needs to be converted into the discrete time domain for the application of the MPC-based collision avoidance. Due to safety concerns, the mobile robot rotational velocity is set to be below a threshold; keeping this in mind and also choosing the system sample time to be small, it is assumed that the robot orientation θ_s is almost constant between the sampling times (T_s). As a result, the discrete-time model will have the following approximate general form:

$$\begin{aligned} &\mathcal{X}(t+(i+1)T_s) = A_d(\theta_s(t+iT_s))\ \mathcal{X}(t+iT_s)\ +\ B_d(\theta_s(t+iT_s))\ U(t+iT_s) \\ &U(t+iT_s) = \begin{bmatrix} f_m(t+iT_s) \\ \dot{c}^d(t+iT_s) \end{bmatrix} \quad i \in [1:k] \end{aligned} \tag{4.6}$$

The index "i" represents the time steps in the rolling horizon window into the future; since the human operator commands are not known in advance, a simple linear extrapolation method based on previous measurements of the force sensor attached to the master haptic device is utilized to estimate the operator future forces $\hat{f}_m(t+iT_s)$ in the prediction horizon of the MPC, i.e. $i \in [1:k]$

$$\begin{aligned} &\hat{\mathcal{X}}(t+(i+1)T_s) = A_d(\theta_s(t+iT_s))\ \hat{\mathcal{X}}(t+iT_s)\ +\ B_d(\theta_s(t+iT_s))\ U(t+iT_s) \\ &U(t+i) = \begin{bmatrix} \hat{f}_m(t+iT_s) \\ \dot{c}^d(t+iT_s) \end{bmatrix} \quad i \in [1:k] \end{aligned} \tag{4.7}$$

where $\hat{\mathcal{X}}(t+iT_s)$ denotes the predicted state vector and the velocity command vectors $\dot{c}^d(t+iT_s)$ are the optimization decision variables. The MPC-based framework utilizes

the predicted states of the system to find optimal values for these decision variables.

Figure 4.1 provides a general overview of the MPC-based collision avoidance technique. In case if the k^{th} step ahead predicted position of the mobile robot $(\hat{x}_s(t+kT_s), \hat{y}_s(t+kT_s))$ may cause collision, the MPC exploits optimization variables $\dot{c}^d(t+iT_s)$ to avoid obstacles by correcting the predicted mobile robot path.

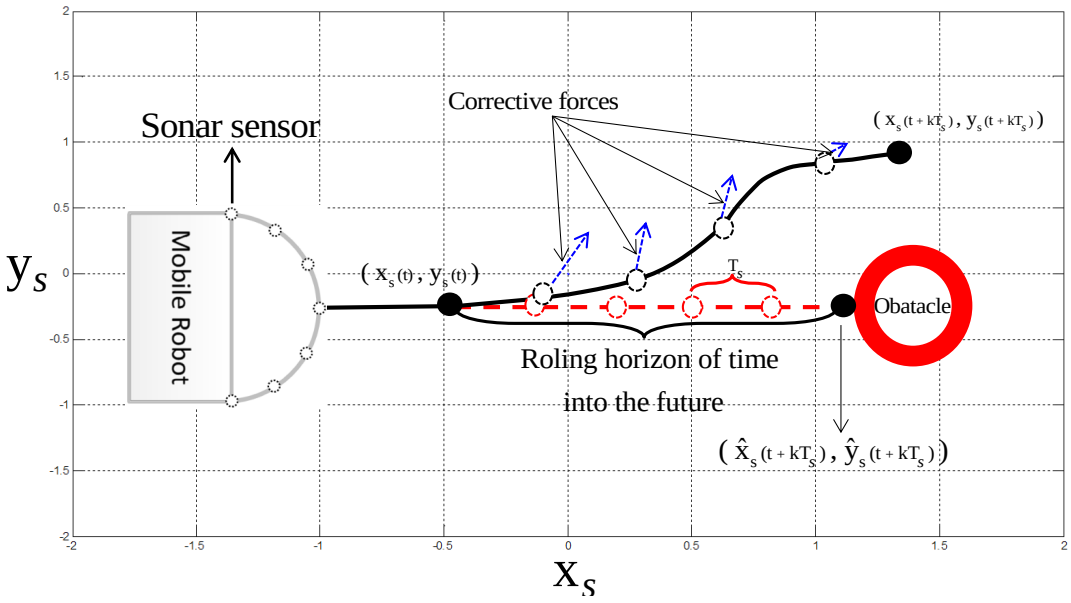

Figure 4.1: Red dashed path is the predicted position of mobile robot in a rolling horizon of time and the solid black one is the real position of the mobile robot after being corrected by the forces of the autonomous collision avoidance subtask. The hollow circles represent mobile robot position at each sample time.

The goal is to avoid the obstacles detected by the mobile robot sonar sensors while minimizing interference with the operator teleoperation actions. Figure 4.2 shows a block diagram of the overall control system.

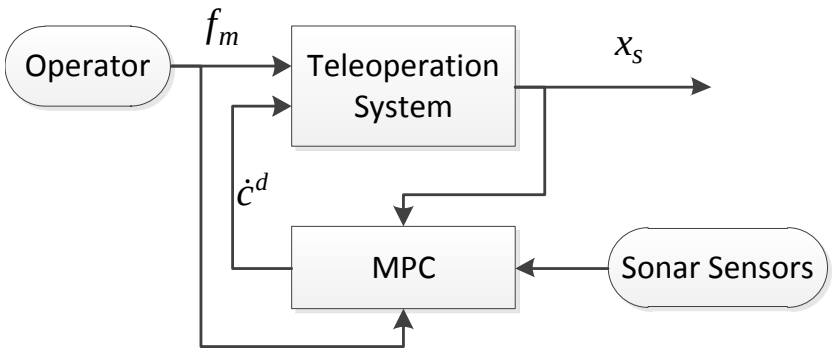

Figure 4.2: Teleoperation control with MPC-based collision avoidance.

Considering detected obstacles and also future predicted operator forces $f_m(t + iT_s), i \in [1 : k]$, the MPC block finds the optimized solution for decision variables $\dot{c}^d(t + iT_s), i \in [1 : k]$ in such way that the operator would feel as small interference as possible and also collision avoidance be guaranteed during the specified rolling horizon time. Therefore, the optimization problem has the following form:

$$\min_{\dot{c}^d(t+iT_s)} \quad H(\hat{f}_m(t + iT_s), \dot{c}^d(t + iT_s)) \tag{4.8}$$

subject to :

$$\hat{\mathcal{X}}(t + (i + 1)T_s) = A_d(\theta_s(t + iT_s))\hat{\mathcal{X}}(t + iT_s) + B_d(\theta_s(t + iT_s)) \begin{bmatrix} \hat{f}_m(t + iT_s) \\ \dot{c}^d(t + iT_s) \end{bmatrix} \tag{4.9}$$

$$\hat{\mathcal{X}}(t + iT_s) \in M \qquad i \in [1 : k] \tag{4.10}$$

$$\left\| \left[\hat{\dot{x}}_s(t + iT_s), \hat{\dot{y}}_s(t + iT_s) \right] \right\|_2 < TV - Threshold \tag{4.11}$$

$$|\hat{\dot{\theta}}_s(t + iT_s)| < RV - Threshold \tag{4.12}$$

$$|\hat{f}_m(t + iT_s) + f_c(\dot{c}^d(t + iT_s))| < Force - Threshold \tag{4.13}$$

where H is the interference cost and is a function of $\hat{f}_m(t + iT_s)$ (human operator

predicted forces) and $\dot{c}^d(t+iT_s)$ (decision variables) , $\hat{\mathcal{X}}$ are predicted system states, M is the obstacle free space, and A_d, B_d are discretisized form of the A, B matrices described by Equation (4.4) calculated as:

$$A_d(\theta_s(t+iT_s)) = e^{A(\theta_s(t+iT_s))T_s} \tag{4.14}$$
$$B_d(\theta_s(t+iT_s)) = \int_0^{T_s} e^{A(\theta_s(t+iT_s))\lambda} d\lambda \ . \ B(\theta_s(t+iT_s))$$

Equation (4.9) represents constraints on the state vector in the form of discrete-time state-space model and (4.10) are the constraints that would ensure a collision-free operation of the mobile robot in the prediction window.

Equations (4.11) & (4.12) are related to the mobile robot translational and rotational velocities respectively. The constraint in (4.13) represents the maximum force of the haptic device, where f_c is the virtual force caused by $\dot{c}^d$ derived in (3.24).

4.2.2 Obstacle Avoidance Constraints

Constraints related to the local obstacles are calculated based on the information received from a 2D range-finder sensor mounted on the vehicle. In the particular system used in this thesis, the robot can detect obstacles using sonar sensors from eight different angles ($N_s = 8$) as depicted in Figure 4.3. As shown in Figure 4.4, the obstacles coordinates are calculated using sonar sensor data, mobile robot position and orientation as

$$O(t,j) = \begin{bmatrix} x_o(t,j) \\ y_o(t,j) \end{bmatrix} = \begin{bmatrix} x_s(t) \\ y_s(t) \end{bmatrix} + S_j \begin{bmatrix} \cos\left(\theta_s(t) + \alpha_j\right) \\ \sin\left(\theta_s(t) + \alpha_j\right) \end{bmatrix}, \quad j \in [1 : N_s] \tag{4.15}$$

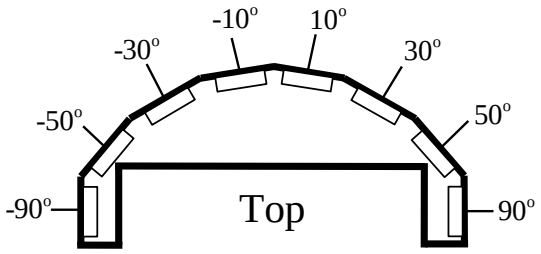

Figure 4.3: P3-DX sonar array.

where $O(t, j)$ is the observed obstacle position at time "t" by j^{th} sonar sensor and α_j is the j^{th} sonar sensor angle with respect to the mobile robot.

So, at each sample time, N_s points are detected as:

$$\begin{bmatrix} x_o(t) \\ y_o(t) \end{bmatrix} = \begin{bmatrix} x_o(t,1), ..., x_o(t,N_s) \\ y_o(t,1), ..., y_o(t,N_s) \end{bmatrix} \tag{4.16}$$

The last n_{sample} detected points are saved in a concatenated form as:

$$\begin{bmatrix} x_o(j) \\ y_o(j) \end{bmatrix} = \begin{bmatrix} x_o(t-(n_{sample}-1)T_s), ..., x_o(t-T_s),\ x_o(t) \\ y_o(t-(n_{sample}-1)T_s), ..., y_o(t-T_s),\ y_o(t) \end{bmatrix},\ j \in [1 : N_s \times n_{sample}] \tag{4.17}$$

A simple form to describe collision avoidance constraints is to keep future mobile robot positions sufficiently away from detected points. One may derive the following

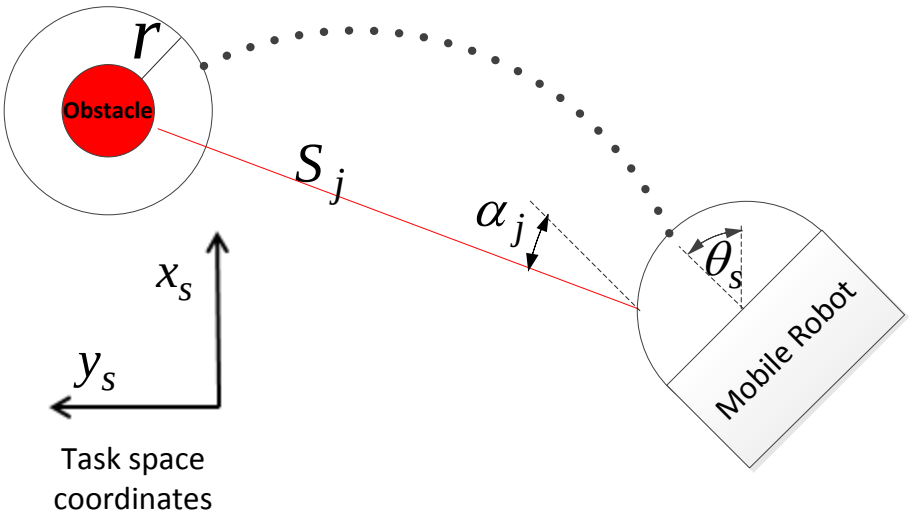

Figure 4.4: Obstacle coordinates in task space frame.

inequality as the collision avoidance constraints in (4.10):

$$\left\| \begin{bmatrix} \hat{x}_s(t+iT_s) \\ \hat{y}_s(t+iT_s) \end{bmatrix} - \begin{bmatrix} x_o(j) \\ y_o(j) \end{bmatrix} \right\|_2 > r \tag{4.18}$$

$$i \in [1:k], \quad j \in [1: N_s \times n_{sample}]$$

where $[\hat{x}_s(t+iT_s), \hat{y}_s(t+iT_s)]^T$ is the estimated mobile robot position at i step ahead into the future and r is the collision avoidance radius. Since the optimization problem should be solved at each sample time and more number of constraints requires more processing time, there would be a limit on the n_{sample} which may cause poor performance in unknown remote environments. A more advance method to represent obstacle constraints in a compact form will be discussed in the next chapter.

4.2.3 Cost Function

The goal here is to minimize the interference of the autonomous collision avoidance subtask velocity command $\dot{c}^d$ with estimated operator desired commands $\hat{f}_m$ over the prediction window in the MPC framework.

Human operator navigates the mobile robot by producing desired force vector f_m and simultaneously feels the autonomous subtask related virtual force vector f_c; notice that this virtual force is a function of $\dot{c}^d$ as derived in Equation (3.24). A larger virtual force results in a bigger interference with the operator; also when the generated virtual force is in direction of the f_m, the operator would feel less interference compared to the case in which f_c is completely in opposite direction of the f_m.

Therefore, one could argue that the interference depends on the norm of $f_c(t+iT_s)$ and also the angle between $f_c(t+iT_s)$ and $\hat{f}_m(t+iT_s)$, denoted as "ν_i". A simple method to derive this angle is through calculating its cosine; and instead of minimizing ν_i, the $\cos\nu_i$ could be maximized. The following function is chosen to quantify the cost of interference:

$$H(\hat{f}_m, \dot{c}^d) \triangleq \sum_{i=1}^{k} w(i) \left[f_c(t+iT_s)^T Q f_c(t+iT_s) - \lambda \cos\nu_i \right] \tag{4.19}$$

$$\cos\nu_i = \frac{f_c(t+iT_s)^T \ \hat{f}_m(t+iT_s)}{||f_c(t+iT_s)||_2 \ ||\hat{f}_m(t+iT_s)||_2}$$

where $w \in \mathbb{R}^{1\times k}$ is a positive gain vector to signify the importance of interference minimization at each sample time and λ is a positive scalar which determines the relative dominance of the norm ($\|f_c\|_2$) and the angle (ν_i) minimization in the cost function. $Q \in \mathbb{R}^{3\times 3}$ is a positive diagonal weighting matrix to form a weighted

norm. In extreme cases, if $Q(1,1)\&Q(2,2) \gg Q(3,3)$, the MPC tries to avoid obstacles by imposing a large torque on the operator to rotate the mobile robot, or if $Q(1,1)\&Q(2,2) \ll Q(3,3)$, the MPC output would be a large force to pull back the mobile robot and decrease its translational velocity.

To implement this function a modification is needed. To avoid a zero denominator in calculating the $\cos\nu_i$, a small constant scalar ϵ is added

$$\cos\nu_i \approx \frac{f_c(t+iT_s)^T \ \hat{f}_m(t+iT_s)}{(||f_c(t+iT_s)||_2+\epsilon)\ (||\hat{f}_m(t+iT_s)||_2+\epsilon)} \tag{4.20}$$

By substituting the (4.20) into the (4.19), the modified cost function becomes:

$$\begin{aligned} H(\hat{f}_m,\dot{c}^d) &\triangleq \sum_{i=1}^{k} w(i)\left[f_c(t+iT_s)^T Q f_c(t+iT_s) - \lambda\cos\nu_i\right] \\ \cos\nu_i &\approx \frac{f_c(t+iT_s)^T \ \hat{f}_m(t+iT_s)}{(||f_c(t+iT_s)||_2+\epsilon)\ (||\hat{f}_m(t+iT_s)||_2+\epsilon)} \end{aligned} \tag{4.21}$$

Figure 4.5 is a simple example where the "b1" and "b2" vectors cause the same interference cost with the fixed vector of "a". The presented locus of vectors are solutions to the following equation:

$$\begin{aligned} & b^Tb - \lambda\frac{b^Ta}{\sqrt{(b^Tb+\epsilon)\ (a^Ta+\epsilon)}} = Cost \\ & a=[1,1]^T,\ \lambda=4,\ \epsilon=10^{-4} \\ & Cost=2 \Longleftrightarrow b1 \\ & Cost=4 \Longleftrightarrow b2 \end{aligned} \tag{4.22}$$

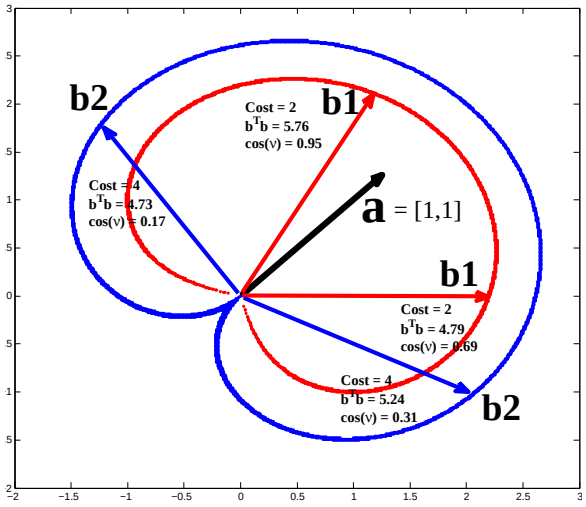

Figure 4.5: A simple 2D example which gives an insight into the proposed interference cost function. The "b1" and "b2" vectors are locus of vectors which have the same interference value with the given fixed vector "a".

4.2.4 Final Form of MPC Optimization Problem Formulation

Among all solutions for $\dot{c}^d$ that are in collision free space and satisfy velocity and force constraints during the prediction horizon, the one that minimizes a measure of interference with operator's commands is chosen. To minimize the interference with the operator teleoperation actions, the norm of virtual force vector $f_c(t + iT_s)$ and also the angle between this vector and the estimated human operator force vector $\hat{f}_m(t + iT_s)$ are forced to be as small as possible. By exploiting Equations (4.21), (4.18), (4.9), (4.11), (4.12), and (4.13), the velocity commands for collision avoidance autonomous subtask over the rolling horizon prediction window are the solution to

the following optimization problem:

$$\min_{\dot{c}^d(t+iT_s)} \sum_{i=1}^{k} w(i) \left[f_c(t+iT_s)^T Q f_c(t+iT_s) - \frac{\lambda f_c(t+iT_s)^T \ \hat{f}_m(t+iT_s)}{(||f_c(t+iT_s)||_2+\epsilon)\ (||\hat{f}_m(t+iT_s)||_2+\epsilon)} \right]$$

subject to :

$$\hat{\mathcal{X}}(t+(i+1)T_s) = A_d(\theta_s(t+iT_s))\hat{\mathcal{X}}(t+iT_s) + B_d(\theta_s(t+iT_s)) \begin{bmatrix} \hat{f}_m(t+iT_s) \\ \dot{c}^d(t+iT_s) \end{bmatrix}$$

$$\left\| \begin{bmatrix} \hat{x}_s(t+i) \\ \hat{y}_s(t+i) \end{bmatrix} - \begin{bmatrix} x_o(j) \\ y_o(j) \end{bmatrix} \right\|_2 > r, \quad j \in [1 : N_s \times n_{sample}]$$

$$\left\| \left[\hat{\dot{x}}_s(t+iT_s), \hat{\dot{y}}_s(t+iT_s) \right] \right\|_2 < TV - Threshold$$

$$|\hat{\dot{\theta}}_s(t+iT_s)| < RV - Threshold$$

$$|\hat{f}_m(t+iT_s) + f_c(\dot{c}^d(t+iT_s))| < Force - Threshold \tag{4.23}$$

Notice that, estimated teleoperation system state $\hat{\mathcal{X}}$, consists of $\hat{X}_s(t+i)$, the i step ahead predicted TCF coordinates of the slave robot, and its derivatives, i.e.

$$\hat{\mathcal{X}}(t+iT_s) = \begin{bmatrix} \hat{X}_s(t+iT_s) \\ \dot{\hat{X}}_s(t+iT_s) \end{bmatrix}, \quad \hat{X}_s = \begin{bmatrix} \hat{x}_s \\ \hat{y}_s \\ \hat{\theta}_s \end{bmatrix} \tag{4.24}$$

The initial values for slave robot TCF coordinates and its derivatives, i.e. $X_s(t+0)\&\dot{X}_s(t+0)$, are obtained using the signals received from encoders attached to the mobile robot wheels. Thereafter, the predicted states $\hat{\mathcal{X}}(t+(i+1)T_s)$ would be based on the slave robot TCF coordinates and its derivatives at time zero and also the predicted human operator force vector $\hat{f}_m(t+iT_s)$ in the rolling horizon window.

The solution to the optimization problem in (4.23) yields the optimal values $\dot{\mathfrak{C}}^d(t+i)$ for future time steps; however only the solution for the first sample time would be implemented (used in Figure 4.2), i.e.

$$\dot{c}^d = \dot{\mathfrak{C}}^d(t+1) \tag{4.25}$$

For the next sample time, the horizon window is moved one sample time ahead and the optimization problem is solved again.

Chapter 5

Environment Feature Extraction

In Chapter 4, the raw measurements provided by 2D range-finder sonar sensor were directly transformed to X-Y coordinates and were used in the formulation of obstacle related constraints. Collision with obstacles were avoided by ensuring that future mobile robot positions were sufficiently away from these obstacle points. During the navigation, the number of obstacle points progressively increases and as mentioned before, the value of n_{sample} in Equation (4.18) determines how many of the previously detected points are utilized. Choosing a large number for n_{sample}, results in a more comprehensive understanding of the remote environment but demands more computational time for the optimization problem to be solved.

The proposed algorithm in this Chapter provides a compact form for obstacle related constraints by focusing on indoor structured environments and assuming that the task space environment can fully be modeled by line segments. By representing obstacles in a compact form with line segments instead of a huge number of individual points, this method requires much less storage and computational time compared with the point-based approach presented in Chapter 4. Moreover, since the obstacle points

are converted to robustly detected line segments, the probability of noise propagation decreases significantly and noise measurements will be discarded automatically. It should be noted that the algorithm presented in this chapter is only applicable to environments with stationary obstacles; inclusion of moving obstacles in the collision avoidance assistance remains outside the scope of current work and will be subject for future research.

The goal of the feature extraction algorithm is to use the 2D range data from the sonar sensors to find an unknown number of walls locally positioned around the mobile robot and specify the position and orientation of detected walls. The list of detected walls is continuously updated based on real-time sensor information and is used by the MPC collision avoidance algorithm.

The Hough transform is one of the most popular methods for line segment extraction and is capable of robustly determining positions and orientations of the walls in the environment. In this chapter, the Hough transform technique as a tool for environment feature extraction is briefly reviewed and then integrated into new formulation for MPC-based obstacle avoidance algorithm.

5.1 Hough Transform

As depicted in Figure 4.3, the mobile robot is equipped with an ultrasonic sonar array which provides a 2D view of the task space environment. Two Cartesian coordinate frames need to be defined: A stationary *"world coordinate frame"* as well as a *"robot coordinate frame"* which is attached to the robot, i.e. see Figure 5.1. Each data from

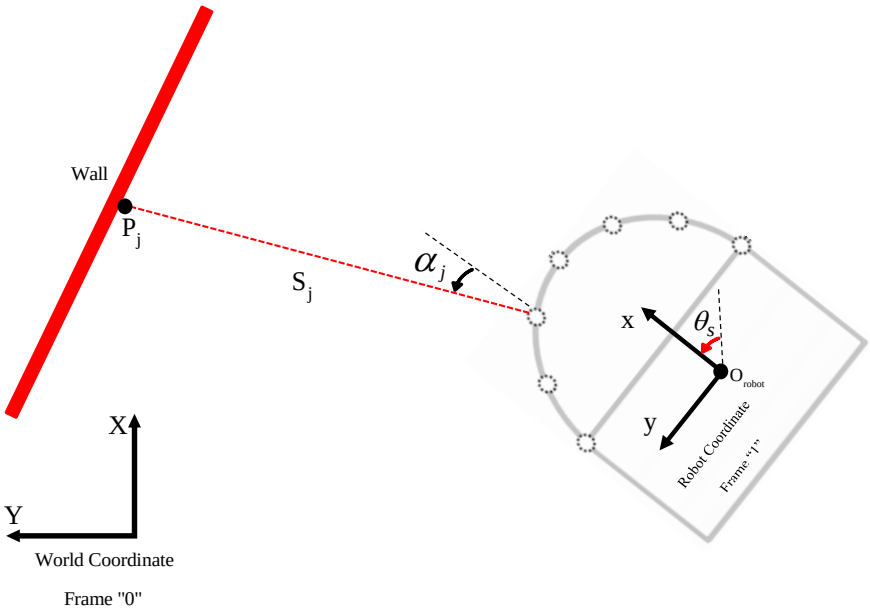

Figure 5.1: Cartesian representation of a detected point.

the sonar sensors is described in the world coordinate frame using:

$$P^0(j) = R_1^0 P^1(j) + O_{robot}^0 \tag{5.1}$$

where P^1 is the coordinates of detected point with respect to the robot coordinate frame (Frame 1), O_{robot}^0 is the origin of robot coordinate frame with respect to the world coordinate frame and R_1^0 is the rotation matrix specifying the orientation of

Frame 1 with respect to Frame 0. These variables can be written as:

$$R_1^0 = \begin{bmatrix} \cos\theta_s & -\sin\theta_s \\ \sin\theta_s & \cos\theta_s \end{bmatrix} \tag{5.2}$$

$$P^1(j) = S_j \begin{bmatrix} \cos\alpha_j \\ \sin\alpha_j \end{bmatrix} \tag{5.3}$$

$$O_{robot}^0 = \begin{bmatrix} x_s \\ y_s \end{bmatrix} \tag{5.4}$$

As depicted in Figure 5.1, θ_s is the mobile robot orientation, α_j is the j^{th} sonar sensor angle with respect to the mobile robot and S_j is the j^{th} sonar sensor measured distance. Therefore the Cartesian coordinates of P_j with respect to the world frame are:

$$P^0(j) = S_j \begin{bmatrix} \cos(\alpha_j + \theta_s) \\ \sin(\alpha_j + \theta_s) \end{bmatrix} + \begin{bmatrix} x_s \\ y_s \end{bmatrix} \tag{5.5}$$

5.1.1 Point-Curve Transformation

The Hough transform is indeed a parametrization method which converts the Cartesian space into the polar space, i.e.

$$\rho = x\cos\theta + y\sin\theta \tag{5.6}$$

It can easily be seen from Equation (5.6) that the set of all lines passing through a fixed point in the Cartesian space (x_c, y_c) would result in a sinusoidal curve in the

"$\rho - \theta$" plane (Figure 5.2).

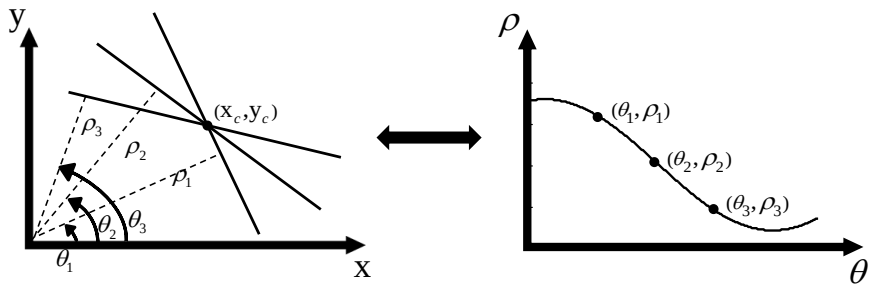

Figure 5.2: All lines passing through a fixed point in the "x-y" plane are transformed to a sinusoidal curve in the "$\rho - \theta$" plane, ($\rho = x_c \cos\theta + y_c \sin\theta$).

Also, a set of sinusoidal curves intersecting at a single point in the polar coordinates (ρ_c, θ_c), are converted to a line in the Cartesian space (Figure 5.3) where ρ_c is the distance from the origin to the line and θ_c is the angle specifying the line orientation.

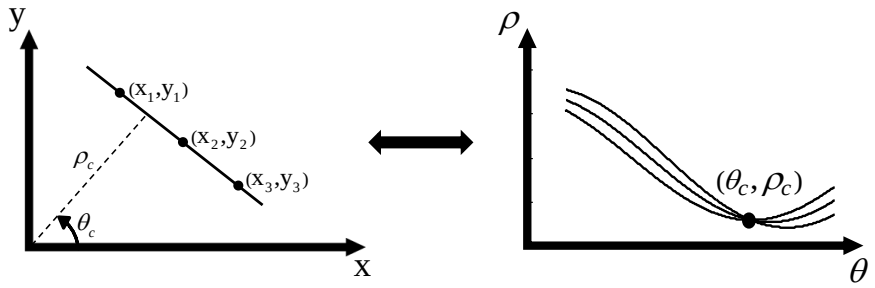

Figure 5.3: A line in the "x-y" plane ($x\cos\theta_c + y\sin\theta_c = \rho_c$) corresponds to infinite number of sinusoidal curves which all intersect at a single point in the "$\rho - \theta$" plane.

The Hough transform converts observed points described in the Cartesian space to corresponding sinusoidal curves in Hough domain, and using the fact that any straight line can uniquely be specified by two parameters (ρ, θ), searches for any possible intersection of these sinusoidal curves.

5.1.2 Recognizing Intersection Points

To find specific points where curves intersect in the Hough domain, several methods have been proposed in the literature. Among these, the most useful techniques reviewed by Glennon (1998) are briefly discussed here.

- ***"Explicitly Solving for Intersections"***

Consider two points in the "x-y" plane; their corresponding curves in the Hough domain are expressed as:

$$\rho = x_1 \cos\theta + y_1 \sin\theta \tag{5.7}$$

$$\rho = x_2 \cos\theta + y_2 \sin\theta \tag{5.8}$$

Since any two points in Cartesian space are collinear, the related curves in the Hough domain must intersect. Solving Equations (5.7) & (5.8) to find the intersection point, one can show that:

$$\rho = (x_1 - x_2)\cos\theta + (y_1 - y_2)\sin\theta = 0 \tag{5.9}$$

$$\Rightarrow$$

$$\theta^* = \arctan\frac{x_2 - x_1}{y_1 - y_2} \tag{5.10}$$

$$\Rightarrow$$

$$\rho^* = x_1 \cos\theta^* + y_1 \sin\theta^* \tag{5.11}$$

In cases where more than two points belong to a line are available and in the presence of measurement noise, solving for each combination of curves expectedly results in

clusters of points near the exact solutions. So using this approach demands another step to cluster these points into unknown number of groups (since the environment is unknown) and then find the best solution for each group. There are many approaches to detect unknown number of clusters. For example, neural network based methods are well-known in this area (Glennon, 1998). Two steps of finding intersection points and clustering them, demand many computations which make the algorithm only work well in off-line applications. In this thesis, we are interested in a fast method requiring modest computations so it could be implemented in real time; explicitly solving for intersections would not be a good choice.

- ***"Resolution Grids"***

This method, which was first introduced by Duda and Hart (1972), divides the Hough domain into grid cells. An accumulator is assigned to each cell to count the number of curves passing through it. An accumulator with a big number indicates that its respective cell most probably contains an intersection point. This intersection point is interpreted as coordinates of a detected line in Cartesian space. For example, the histogram of a simple experiment of detecting a single wall is depicted in Figure 5.4; it can be concluded from this that the wall is likely located at $(.82_m, 18_{deg})$. It should be noted that this method only provides approximate coordinates of the walls and not their exact values. The accuracy of this approximation can be improved by reducing the grid size at the expense of having to deal with a larger number of cells and more computations.

The basic idea behind this method is that, analyzing the whole Hough domain to precisely find all intersection points is a time consuming procedure and may result in significant errors in cases where measure noise is significant. Therefore, the Hough

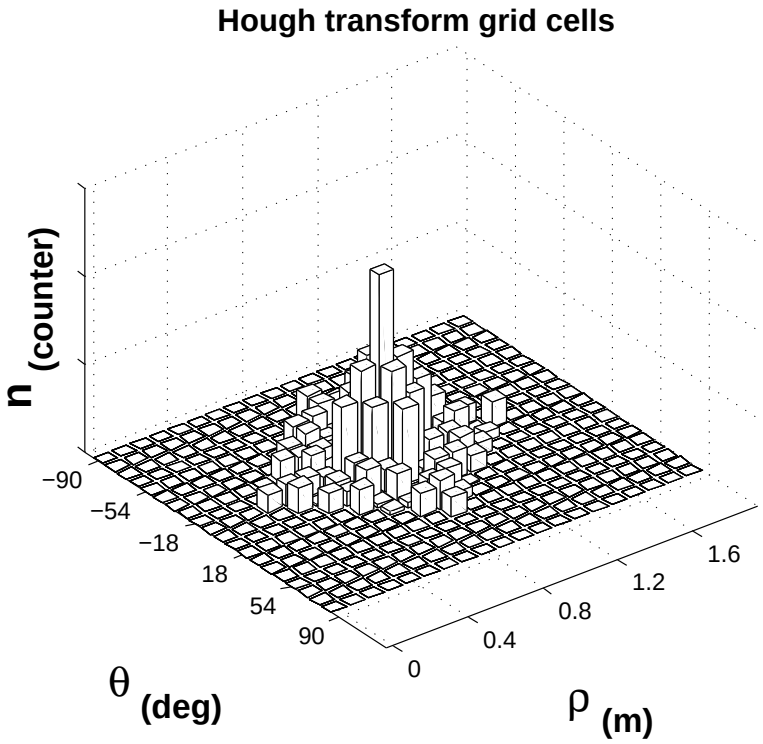

Figure 5.4: Histogram of a simple experiment where the cell with biggest number (n axis) shows the likely coordinates of a line in the Hough space.

space is divided to a two dimensional grid, where the grid resolution can be adjusted to achieve a desired accuracy.

The Grid Resolution technique is summarized in Table 5.1.

In this work, the resolution grid technique is applied to detect intersection points in the Hough domain.

Table 5.1: Steps for Duda and Hart technique (Duda and Hart, 1972; Glennon, 1998).

Step 1	For a given point in x-y coordinates, generate corresponding $\theta - \rho$ curve.
Step 2	Divide the 2D Hough space into a grid; note the cells that the curve crosses and increase the related accumulator by one unit.
Step 3	Repeat steps 1 and 2 for every single point.
Step 4	Extract accumulators that are greater than a predefined threshold.
Step 5	Extract cells related to accumulators derived in Step 4. These cells represent estimation of detected lines coordinates.

5.2 Deriving End Points

Deriving the line coordinates is not enough for the application considered in this thesis. The application of the Hough technique produces a number of pairs in the $(\rho - \theta)$ form, which as depicted in Figure 5.5, specify infinite-length lines. Obviously, this representation can not be a real estimation of the remote environment. The infinite lines should be converted to line segments by finding the end points of the walls, e.g. see Figure 5.5. As shown in Figure 5.3, the line extraction is based on

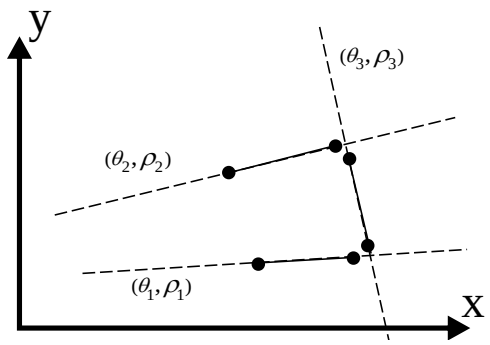

Figure 5.5: A simple example of representing structured environments using infinite-length lines vs. line segments.

converting observed points described in Cartesian space into corresponding Hough domain curves. Hence in order to determine end points for each line, the contributing Cartesian space points are analyzed.

Considering a set of (x,y) pairs forming a detected line (ρ_0, θ_0), the required steps to find the two line end-points, "$LeftEnd$" & "$RightEnd$", are given in 5.2.

Table 5.2: Deriving end points.

Step 1	For a given set of points $\{P_1, P_2, ..., P_n\}$ • Initiate LeftEnd & RightEnd : $LE = P_1$, $RE = P_2$ • Initiate Slack variables : $Rd = (P_2 - P_1).(P_2 - P_1)$ $Ld = 0$
Step 2	**for i = 3 : n** $Prod = (P_i - P_1).(P_2 - P_1)$ **if Prod > Rd** $Rd = Prod$ $RE = P_i$ **elseif Prod < Ld** $Ld = Prod$ $LE = P_i$ **end** **end**
Step 3	$LE\&RE$ are the two end points.

5.3 Final Form of Line Extraction and Segmentation Algorithm

At each sample time, new observed raw points are fed into the proposed algorithm and the stack of stored line segments is updated. For a new point, first it should be checked to see whether it belongs to previously detected line segments or not. If it does, the new point is redundant data and would be discarded. To check this criterion,

we are faced with calculating the distance from the new point $P_{new} = [x_{new}, y_{new}]^T$ to the extracted line segments. Simply, if the calculated distance is more than a predefined threshold, P_{new} does not belong to the line segment.

As shown in Figure A.1, the shortest distance to a line segment depends on the relative location of the specified point with respect to the line segment. First, the point P_{new} is projected onto the line, named as P_n, then according to the location of the P_n, three different distances might be the solution: distance to the first end point (d_1), distance to the second end point (d_2), and distance to the line (d_n).

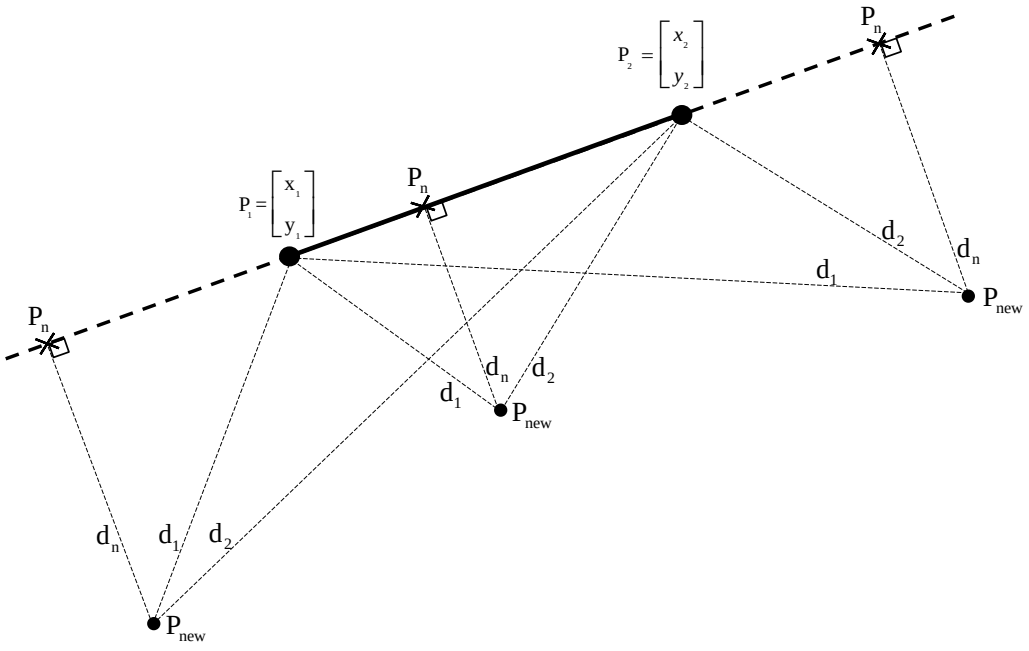

Figure 5.6: Shortest distance between a point and a line segment.

The line equation can be parameterized as follow:

$$rP_1 + (1 - r)P_2 \quad , \quad r \in \mathbb{R} \tag{5.12}$$

As depicted in Figure 5.7, the line is divided into three parts based on the "r" value. So, after finding projected point P_n and then the corresponding "r" value, the shortest distance is calculated as:

$$\text{shortest distance} \triangleq d = \begin{cases} d_1 & \text{for} \quad r > 1 \\ d_n & \text{for} \quad 0 < r < 1 \\ d_2 & \text{for} \quad r < 0 \end{cases} \tag{5.13}$$

As calculated in the appendix, "r" can be explicitly expressed as a function of P_1, P_2 and P_{new}.

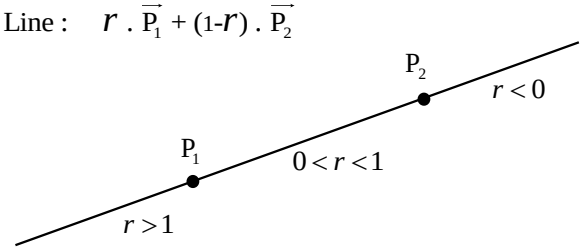

Figure 5.7: Representing an infinite-length line using two points and a single parameter "r".

After checking this condition, if P_{new} does not belong to any of the previous detected line segments, the corresponding sinusoidal curve in the Hough domain is generated and the algorithm searches for crossed cells. For each cell, a memory is

assigned to keep necessary information as:

cell.Acc : accumulator

cell.RE : right end point , cell.Rd : slack variable

cell.LE : left end point , cell.Ld : slack variable

cell.P_1 : first observed point in the cell

cell.P_2 : second observed point in the cell

The final Hough transform-based algorithm which instantaneously updates the stored line segments is given in the Table 5.3. The algorithm output is $[Stack]$ which includes all detected line segments. Two thresholds are defined here, $NThr$ and $LThr$. $NThr$ denotes the number of required points to form a line segment and $LThr$ limits the length of the line segments. If a new point lies within a potential new line, the algorithm (see lines 11 and 19) ensures that by adding this new point to the line segment, its length would not be extended hugely; in other words, if the line segment extension would be greater than $LThr$, this new point would be ignored. This problem usually happens when two separated walls lie on the same line, e.g. office entries.

Table 5.3: Line segment extraction algorithm using Hough transform.

For a new given point in x-y coordinates P_{new}
if $P_{new} \in$ stored line segments in $[Stack]$, **terminate**
else :
  • Generate related $\theta - \rho$ curve, divide the 2D Hough space into a grid with desired resolution along ρ axis "Res_ρ" and θ axis "Res_θ"
  • Note the cells that the curve crosses
  • For each cell:
   if cell.Acc = 0
      cell.Acc = cell.Acc+1
      cell.LE = P_{new}
      cell.P_1 = P_{new}
   elseif cell.Acc = 1
      if $||P_{new} - cell.LE||_2 < LThr$
        cell.Acc = cell.Acc+1
        cell.RE = P_{new}
        cell.P_2 = P_{new}
        cell.Rd = $(cell.P_2 - cell.P_1)^T(cell.P_2 - cell.P_1)$
        cell.Ld = 0
      end
   elseif cell.Acc $\geq$ 2
      if distance from P_{new} to the line segment [cell.LE , cell.RE] $< LThr$
        cell.Acc = cell.Acc+1
        $Prod = (P_{new} - cell.P_1)^T(cell.P_2 - cell.P_1)$
        if $Prod >$ cell.Rd
          cell.Rd = $Prod$
          cell.RE = P_{new}
        elseif $Prod <$ cell.Ld
          cell.Ld = $Prod$
          cell.LE = P_{new}
        end
      end
   end
  • Choose the element with biggest accumulator, $n_{max} = cell^*.Acc$
  if $n_{max} < NThr$, **terminate**
  else :
    • $Stack = [Stack, [cell^*.LE, cell^*.RE]^T]$
    • Clear $cell^*$ associated memories.
  end
end

5.4 Integration of Line Segment-based Constraints into MPC Optimization

To use the Hough transform environment feature extraction technique within the developed MPC optimization model (4.23), the obstacle related constraint should be replaced. The developed algorithm provides the stack of detected line segments; instead of raw points used in the optimization problem (4.23), the stored line segments are employed.

The collision avoidance constraints in Chapter 4 implied that the future predicted mobile robot position $[\hat{x}_s(t+iT_s), \hat{y}_s(t+iT_s)]^T,\ i \in [1:k]$ should not be closer than a predefined radius to observed obstacle points $[x_o(j), y_o(j)]^T,\ j \in [1: N_s \times n_{sample}]$. Replacing the observed points by line segments requires a different measure of distance than the simple euclidean norm defined in Chapter 4.

By utilizing Equation (5.14) and replacing $\{P_1, P_2, P_{new}\}$ by $\left\{LE, RE, [\hat{x}_s(t+i), \hat{y}_s(t+i)]^T\right\}$ respectively, the shortest distance from the i^{th} predicted mobile robot position to the j^{th} line segment, $d(i,j)$ is defined as

$$d(i,j) = \begin{cases} d_1(i,j) & \text{for} \quad r(i,j) > 1 \\ d_n(i,j) & \text{for} \quad 0 < r(i,j) < 1 \\ d_2(i,j) & \text{for} \quad r(i,j) < 0 \end{cases} \tag{5.14}$$

where

$$d_1(i,j) = \left\| \begin{bmatrix} \hat{x}_s(t+iT_s) \\ \hat{y}_s(t+iT_s) \end{bmatrix} - LE(j) \right\|_2 \tag{5.15}$$

$$d_2(i,j) = \left\| \begin{bmatrix} \hat{x}_s(t+iT_s) \\ \hat{y}_s(t+iT_s) \end{bmatrix} - RE(j) \right\|_2 \tag{5.16}$$

$$d_n(i,j) = \frac{|a(j)\hat{x}_s(t+iT_s) + b(j)\hat{y}_s(t+iT_s) + c(j)|}{\sqrt{a^2+b^2}} \tag{5.17}$$

$$a(j) = LE_y(j) - RE_y(j)$$

$$b(j) = RE_x(j) - LE_x(j)$$

$$c(j) = RE_y(j).LE_x(j) - LE_y(j).RE_x(j)$$

where $r(i,j)$ is a function of $LE(j), RE(j), [\hat{x}_s(t+i), \hat{y}_s(t+i)]^T$ given in the appendix. By using the step function $u(.)$, Equation (5.14) is simplified and the collision avoidance constraints are given as:

$$d(i,j) > \mathfrak{r}^2 \qquad i \in [1:k], j \in [1:n_l] \tag{5.18}$$

$$\begin{aligned} d(i,j) \;=\; & d_1(i,j).u(r(i,j)-1) \;+\; d_2(i,j).u(-r(i,j)) \\ & +\; d_n(i,j).(u(r(i,j)) - u(r(i,j)-1)) \end{aligned}$$

where n_l is the number of detected line segments.

Since the step function is not differentiable at zero, it can not be used in the optimization programming. As shown in Figure 5.8, the following smooth differentiable

function is a close approximation of the step that could be used in optimization.

$$\hat{u}(r) = 0.5 + \left(\frac{1}{\pi}\right) \arctan\left(100\ r\right) \tag{5.19}$$

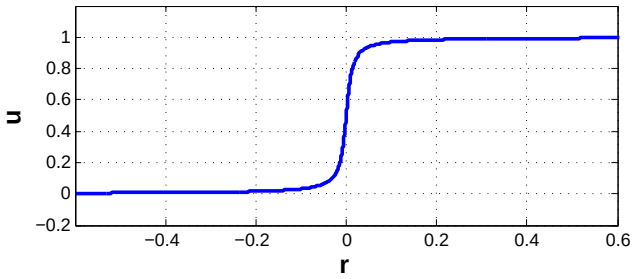

Figure 5.8: Approximated step function for use in optimization.

Chapter 6

Experimental Results

The master/slave system shown in Figs 3.4 and 3.5 was employed in the experiments. The slave is a P3-DX mobile robot equipped with eight(8) sonar sensors at the front to detect obstacles. The master device is a Quanser planar pantograph equipped with ATI Nano25 F/T sensor to measure the end effector forces.

The planar pantograph has three(3) DOF, (x_m, y_m, θ_m), which control the corresponding slave side TCF coordinate variables $X_s = [x_s, y_s, \theta_s]^T$. At the master side, an adaptive low-level velocity controller from Malysz (2011) was used to achieve joint level velocity tracking. The Pioneer P3-DX has its own onboard velocity controller. The real-time code ran under Matlab RTW/Simulink with Quanser Quarc 2.0 with a custom interface between the API of the Pioneer P3-DX and Simulink. The teleoperation control loop operated at a sampling rate of 1 kHz.

The sonar detection range is $1.7(m)$ and the robot maximum velocity is $1(m/s)$. The Matlab function *fmincon* is used to solve the optimization problem in real time. The time steps in the rolling window are 0.2 sec each and the prediction window length is set to five steps, i.e. 1 sec.

First, results for the controller in Chapter 4, with individual point-based representation of the obstacles are given. Next experimental results for the version of the controller that employs line segments models of the obstacles in Chapter 5 are presented.

The control parameters used in these experiments are given in Table 6.1.

Table 6.1: Control parameters

a	$diag(0.1, 0.1, 10)\ kg^{-1}s$	Λ	$I\ s^{-1}$
k_p	$diag(15, 15, 2)$	W_{xc}	I
Ω	$diag(0.2, 0.2, 0.2)$	r	$0.15\ m$
Σ	I	C	$100\pi\ rad/s$
w	$[1, 1, 1, 1, 1]$	λ	0.4
$Force - Threshold$	$[5\ N, 5\ N, 2\ Nmm]^T$	T_s	$0.2\ s$
$TV - Threshold$	$1\ m/s$	k	5
$RV - Threshold$	$5\ rad/s$	Q	$diag(1, 1, .1)$

6.1 Experiment with MPC collision Avoidance and Point-based Representation of Obstacles

As shown in Figure 6.1, in this experiment the human operator guides the mobile through a narrow corridor. The blue circles in Figure 6.2 are mobile robot positions during the experiment and red dots are sonar sensors observed points.

In this experiment $n_{sample} = 30$ which means obstacles detected in the last $30 \times T_s = 6s$ are included at each sample time for MPC-based collision avoidance.

Figure 6.3 shows the master device forces f_m and virtual forces caused by autonomous control subtask $f_c = a^{-1}P_{uxx}^{-1}P_{uxc}\dot{c}^d$ in Equation (3.24). The MPC-based controller tries to keep both virtual forces (see Figure 6.3) and the angle between

Figure 6.1: Task environment in MPC-based collision avoidance experiment with point-based representation of the obstacles.

these force vectors and the human operator generated force vectors (see Figure 6.4) small while preventing potential collisions with the walls. There are times, however, that this angle cannot be small due to the location of the obstacle with respect to the robot. From Figure 6.2, it can be seen that around $t = 14.5sec$, the robot approaches an obstacle, so a large corrective action is required to avoid collision, i.e. see the circled data in Figures 6.3, 6.4 and 6.5. Here the MPC-based controller finds it necessary to push back the robot at the cost of more interference with the operator. Figure 6.3 shows that relatively large virtual forces were produced around this time, which were not in the direction of the desired commands by the human operator, as is evident in Figure 6.4.

Figure 6.5 demonstrates good teleoperation tracking performance where the master and slave TCFs follow each other. The autonomous collision avoidance subtask does

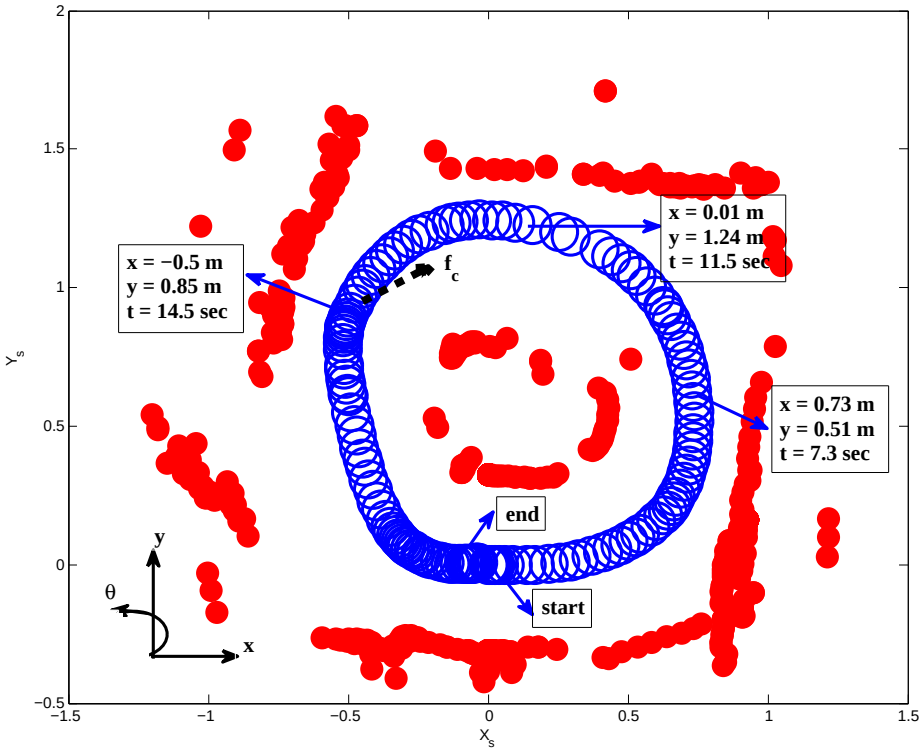

Figure 6.2: Human operator navigates the mobile robot (blue circles) in a narrow corridor where red dots represent sonar sensor measurements. Dashed black arrow is a 2D vector including the first two elements of the 3D virtual force vector caused by collision avoidance autonomous subtask control f_c.

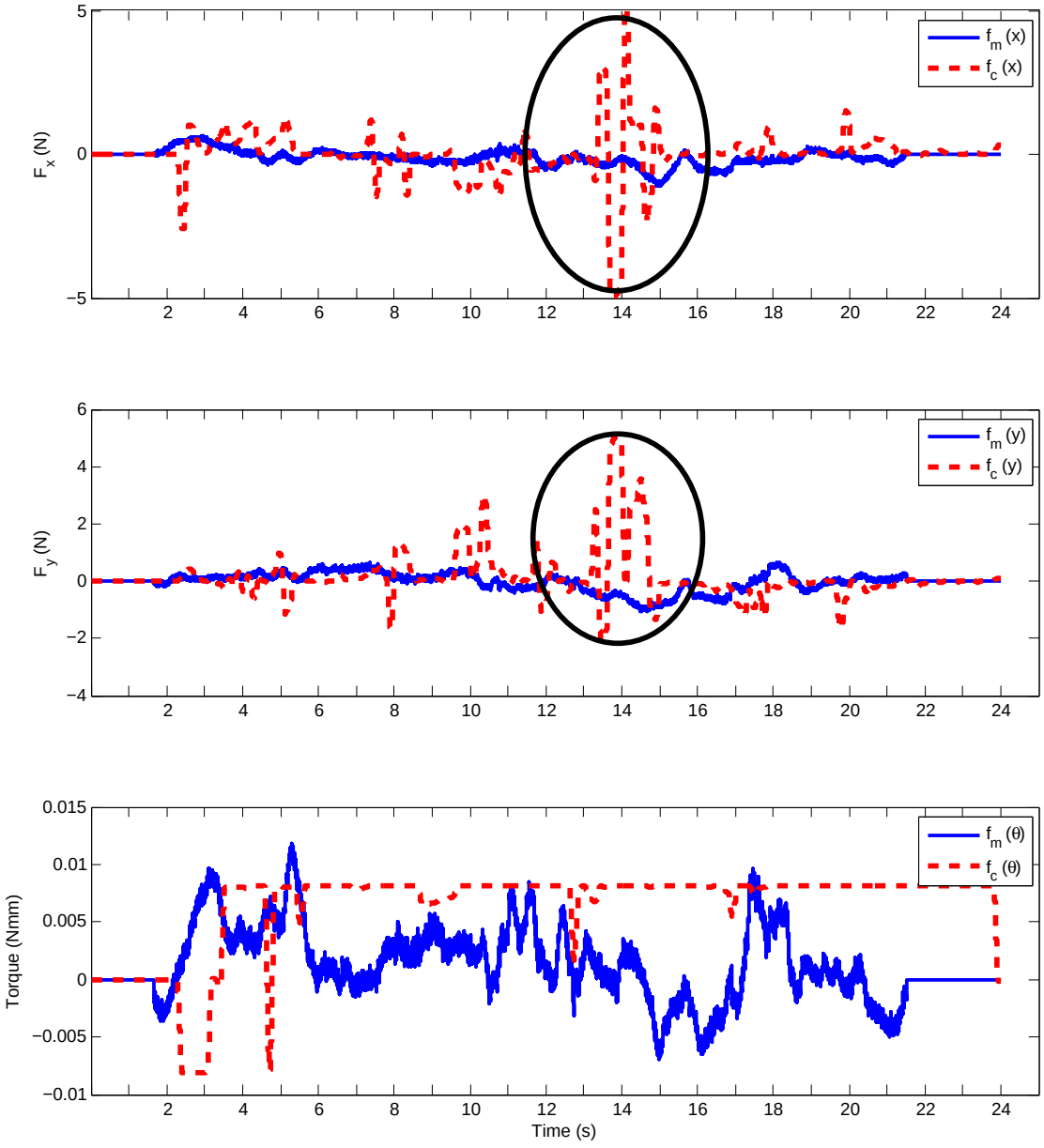

Figure 6.3: Measured operator forces (solid lines) and virtual forces caused by concatenated autonomous control subtask (dashed lines).

not affect position/velocity tracking between master and slave TCFs, but rather provides the operator with proper corrective force feedback to push the robot trajectory away from the obstacles, e.g. see the circled area in Figure 6.5.

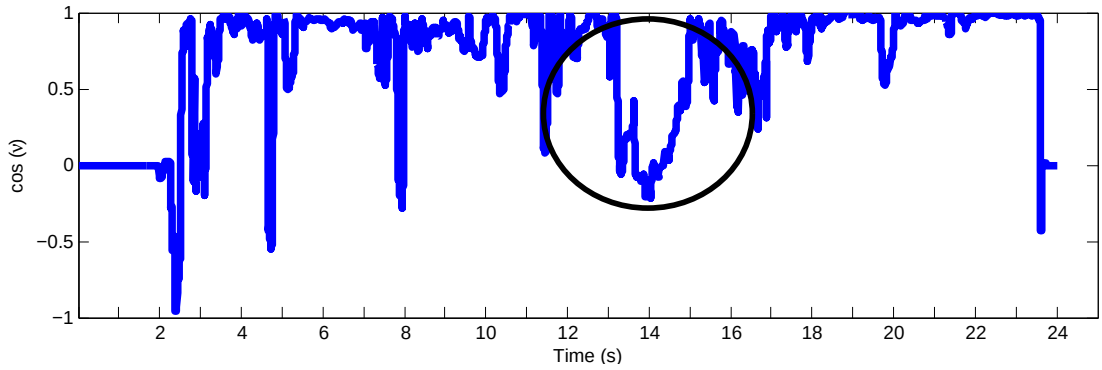

Figure 6.4: Interference measurement, i.e. cosine of the angle between the operator force f_m and correction force f_c.

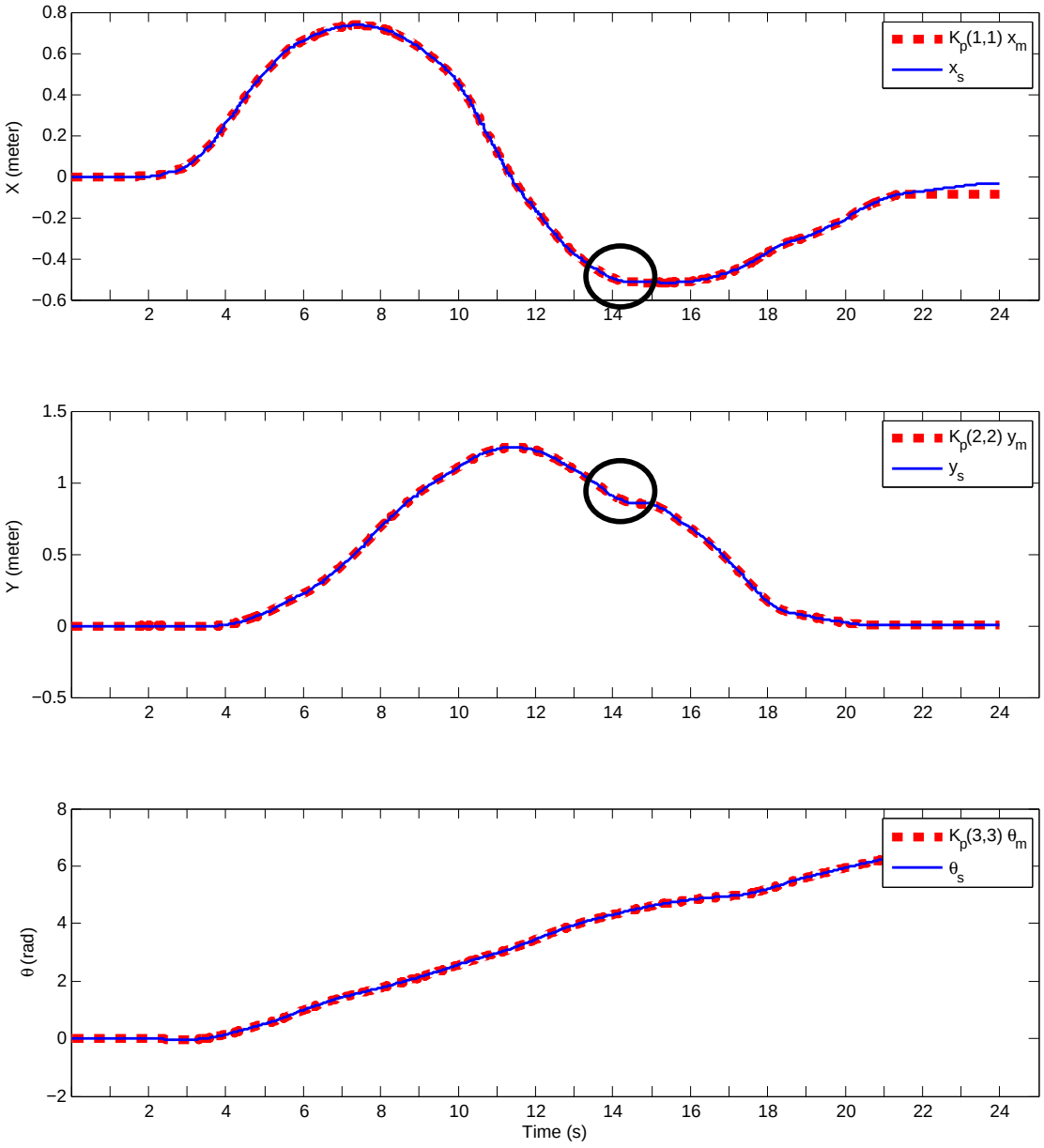

Figure 6.5: Position tracking; master robot sets desired mobile robot position and orientation.

6.2 Experiments with MPC Collision Avoidance and Line Segment-based Representation of Obstacles

Two experiments are presented here where the proposed feature extraction algorithm in Chapter 5 has been utilized by MPC collision avoidance algorithm. Figure 6.6 gives a snapshot of the task environment which is constructed by board segments made of cardboard. The feature extraction related parameters are given in Table 6.2.

Table 6.2: Feature extraction related parameters

$NThr$	10	Res_ρ	4 cm
$LThr$	0.6 m	Res_θ	2 deg

Figure 6.6: Task environment in MPC based collision avoidance experiments with line segment representation of the environment.

The first experiment involved a simple scenario of moving towards a corner. The red dots in Figure 6.7 are sonar sensor observations and the solid black line segments have been constructed by the feature extraction algorithm. The line segments are

detected in a real-time manner to update collision avoidance related constraints. As shown in Figure 6.8, a possible collision in $t = 13.5$ was avoided by generating a force vector in opposite direction of the mobile robot motion and also a torque resulting in a right turn. Figure 6.9 shows a relatively large interference at this special moment. Figure 6.10 displays tracking between master and slave positions around this time and trajectory modification due to the collision avoidance algorithm.

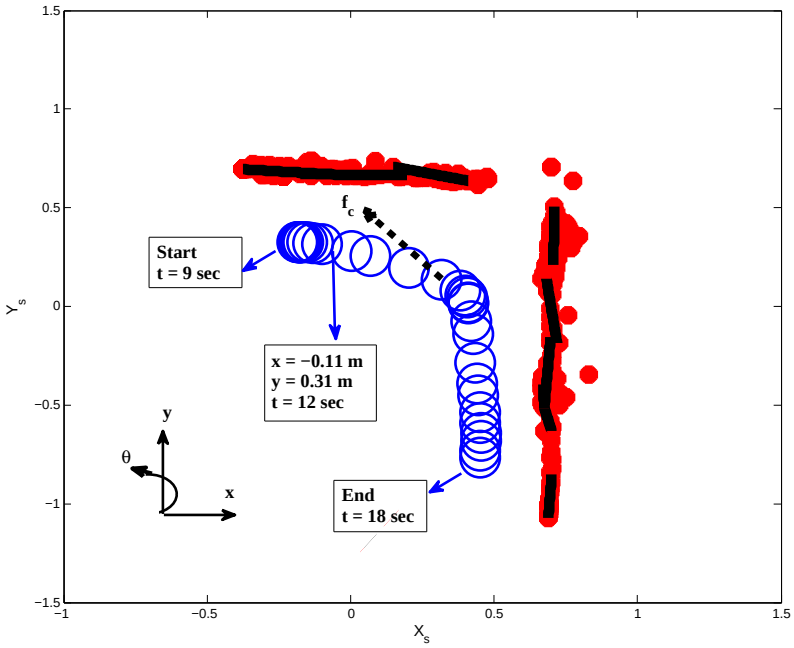

Figure 6.7: First experiment with MPC collision avoidance and line segment-based representation of obstacles; Human operator navigates the mobile robot (blue circles). Red dots are sonar sensor measurements and black line segments representing the environment. Dashed black arrow is a 2D vectors including the first two elements of 3D virtual force vector caused by collision avoidance autonomous subtask control f_c.

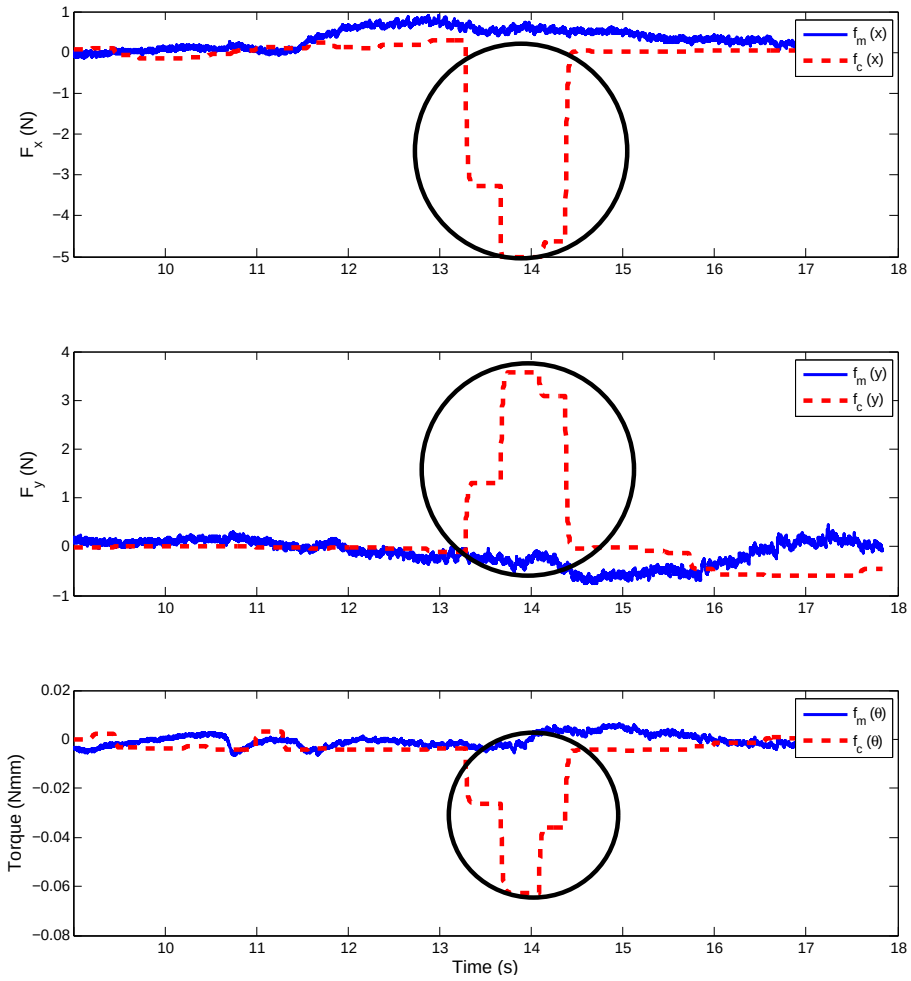

Figure 6.8: First experiment with MPC collision avoidance and line segment-based representation of obstacles; measured operator forces (solid lines) and virtual forces from the collision avoidance algorithm (dashed lines).

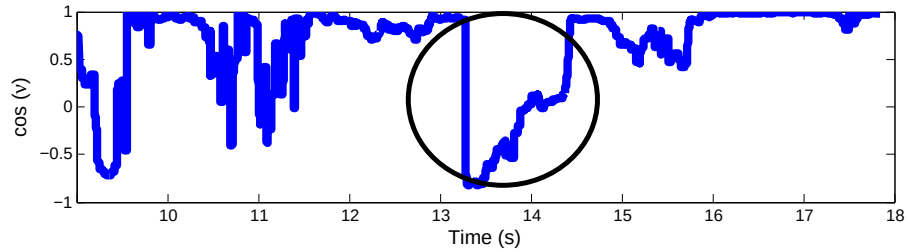

Figure 6.9: First experiment with MPC collision avoidance and line segment-based representation of obstacles; interference measurement, i.e. $\cos(\nu)$.

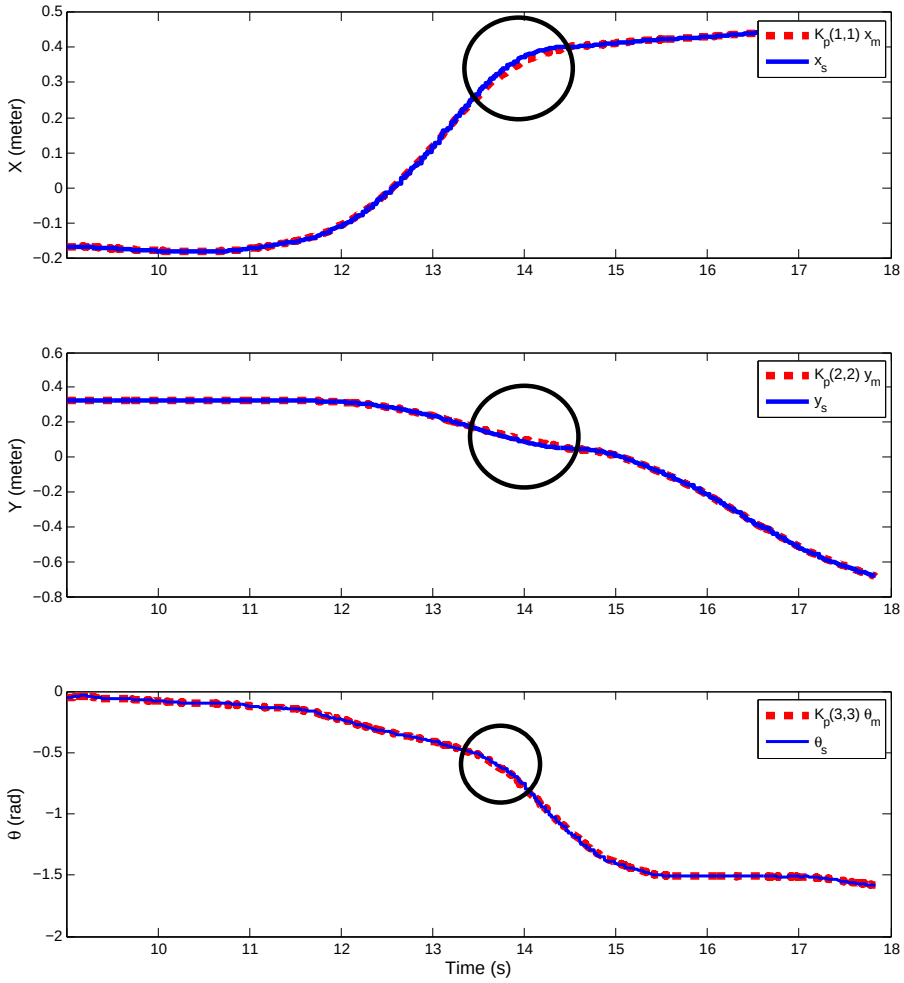

Figure 6.10: First experiment with MPC collision avoidance and line segment-based representation of obstacles; position tracking.

Figure 6.11 shows the position of mobile robot shifted ahead 5 sample times (solid lines) and the predicted position of the mobile robot 5 step ahead (dashed lines).

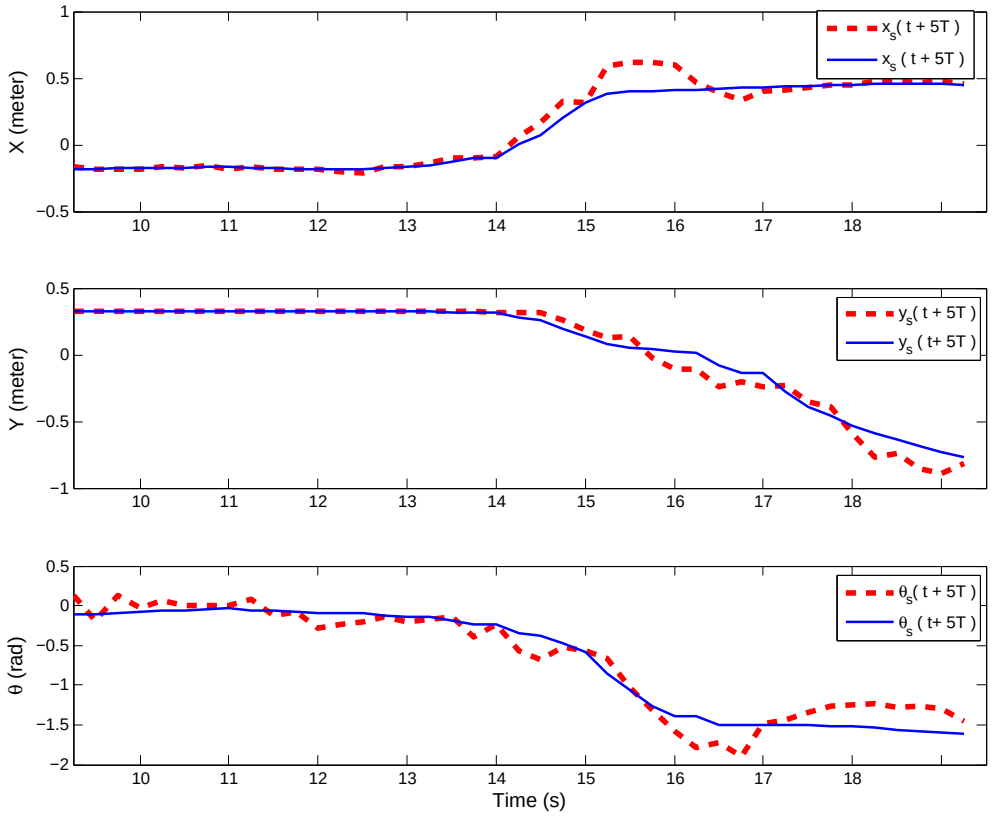

Figure 6.11: First experiment with MPC collision avoidance and line segment-based representation of obstacles; predicted position of the mobile robot 5 step ahead (dashed lines) versus real position of the mobile robot shifted forward by 5 sample times (solid lines).

The second experiment was more general to investigate different possible situations simultaneously. The red dots in Figure 6.12 are sonar sensor observations; as it can be seen, despite the poor quality of sonar sensors detected points, the feature extraction algorithm, provides an acceptable map (solid black line segments).

Around $t = 11sec$, a possible collision was predicted and an opposing force (circled data in Figure 6.13) moved the mobile robot back, which resulted in more interference as depicted in Figure 6.14. Since the backward movement could have caused collision, a negative virtual torque at $t = 16sec$ (circled virtual torque data in Figure 6.13) kept the robot away from the adjacent wall. Master and slave measured positions around this time in Figure 6.15 show the trajectory modification due to the collision avoidance algorithm.

Around $t = 38sec$, another possible collision was predicted and as is evident in Figure 6.13, generated virtual forces along all three axes assisted the human operator to push the mobile robot back which resulted in trajectory modification and accordingly more interference, i.e. see the circled areas in Figures 6.13, 6.14 and 6.15. Since the robot moved back, a negative virtual torque at $t = 42sec$ generated in order to keep it away from the corner. At the end, the robot was moving towards a wall (line segment) and a small force prevented it from colliding with that wall.

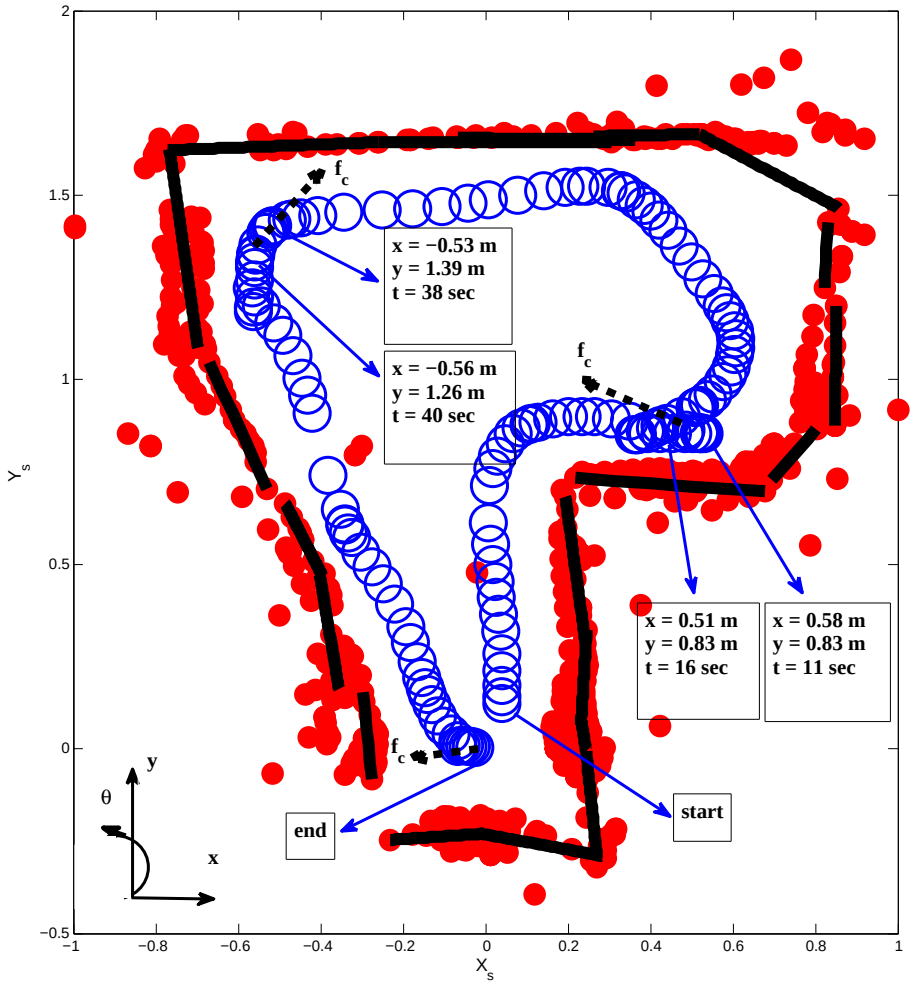

Figure 6.12: Second experiment with MPC collision avoidance and line segment-based representation of obstacles; blue circles are mobile robot positions, red dots are sonar sensor measurements and black line segments representing the environment. Dashed black arrows are 2D vectors including the first two elements of 3D virtual force vectors caused by collision avoidance autonomous subtask control f_c.

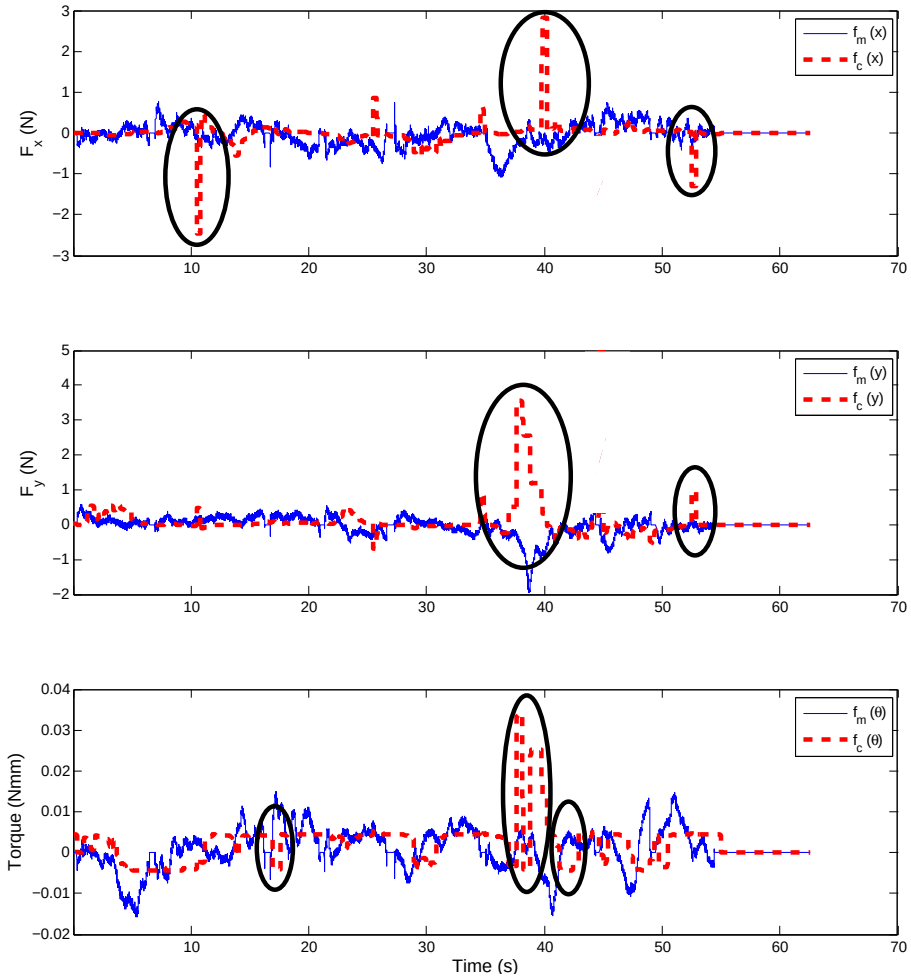

Figure 6.13: Second experiment with MPC collision avoidance and line segment-based representation of obstacles; measured operator forces (solid lines) and virtual forces from the collision avoidance algorithm (dashed lines).

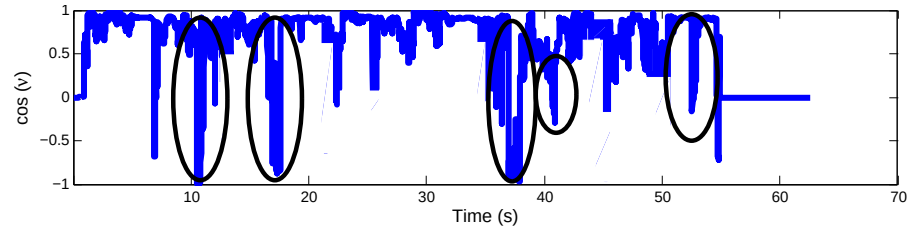

Figure 6.14: Second experiment with MPC collision avoidance and line segment-based representation of obstacles; interference measurement, i.e. $\cos(\nu)$.

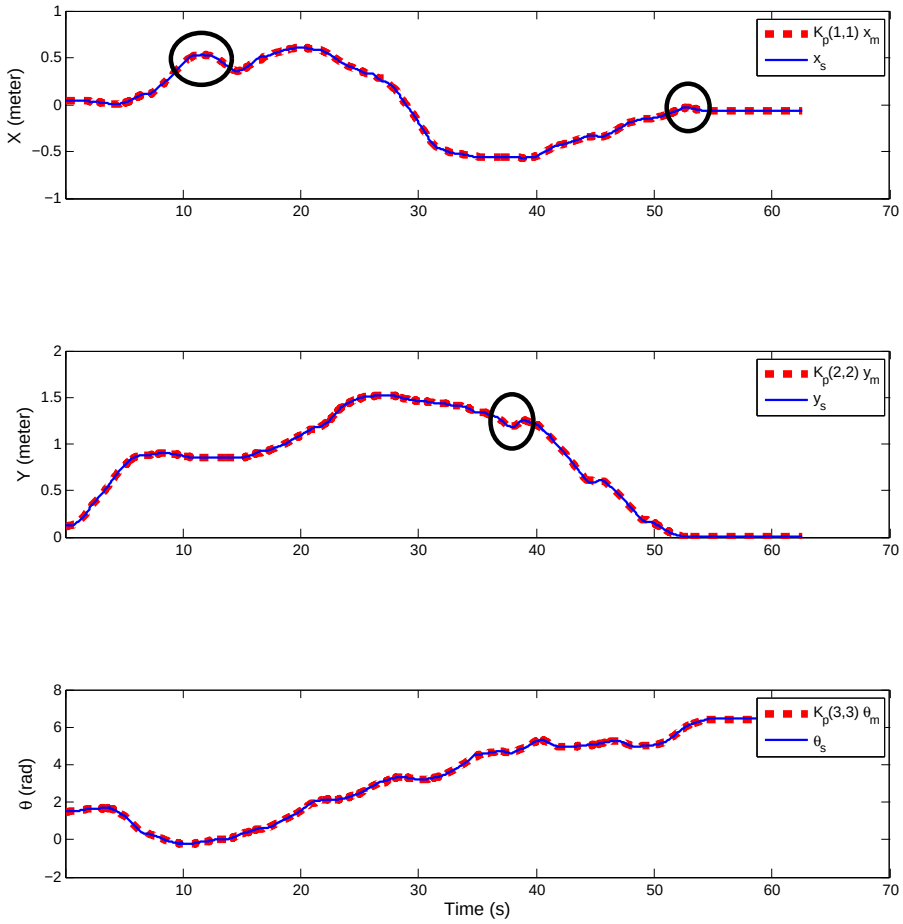

Figure 6.15: Second experiment with MPC collision avoidance and line segment-based representation of obstacles; position tracking.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

This thesis presented a novel method for semi-autonomous teleoperation of a mobile robot in an unknown task environment. The main contribution of this work was the development of an obstacle avoidance method for mobile robot teleoperation systems, where the concept of MPC for collision avoidance was integrated into the general teleoperation control framework of Malysz and Sirouspour (2013). This control framework allows for teleoperation and autonomous subtasks alongside each other with shared priority. An autonomous subtask was defined to help the operator avoid collisions with obstacles.

In the proposed system, the operator navigated a nonholonomic mobile robot using a 3DOF haptic device. The autonomous subtask provided corrective force-type feedback signals aimed at avoiding collision with obstacles detected by on-board sonar sensors. The corrective feedback signals were obtained using an MPC control strategy by formulating and solving a constrained optimization problem over a rolling horizon

window. The aim of optimization problem was to avoid local obstacles with minimum interference with operator's desired teleoperation commands.

Two approaches were proposed to formulate collision avoidance related constraints in the optimization problem: point-based and line segment-based representation of the obstacles. In the point-based approach, the sonar sensors data was directly used in defining collision avoidance related constraints. The main advantage of this approach compared to the other one was its functionality in unstructured environments. However, since raw points were not accurate and progressively increasing by time, a more advance technique based on representing obstacles using line segments introduced to utilize sonar sensors raw data and accurately extract environment features in a more compact form. Experiments in teleoperation of a mobile robot demonstrated the effectiveness of the proposed teleoperation/collision avoidance strategy; where a 3DOF planar Pantograph was employed by human operator to navigate a P3-DX mobile robot equipped with eight (8) sonar sensors at the front.

7.2 Future Work

This thesis integrated MPC-based collision avoidance assistance into the teleoperation of remote mobile robots in order to assist the human operator and reduce the complexities of navigation. While the presented results were encouraging, there are still a number of possibilities for future research, including the ones suggested below.

- The current work assumes stationary obstacles. In the future, a variant of the proposed control approach could be developed to deal with both stationary and moving obstacles.

- The proposed algorithm in Chapter 5 for environment modeling relied on finite length line segment representation. This algorithm could be revised to allow for a broader representation of the environment, i.e. using curved paths, etc.

- In this work, the mobile robot location is calculated using wheel odometery and pulse signals from incremental encoder senors, and the relative location of obstacles is determined using sonar sensors. Considering slippery remote environments and sonar sensors noise measurements, there could always exist uncertainties in relative location of obstacles and mobile robot which would result in collision. A conservative solution is to choose a large value for the mobile robot safety radius; to decrease the conservativeness, the robust optimization could be used to address uncertainty issues. Also, due to uncertainties, strictly imposing collision avoidance constraints might not be appropriate, instead the optimization problem could also be in a chance-constraint programming form; that is by defining constraints in such a way that the probability of the collision is limited to a predefined threshold.

- There might be situations in which the optimization problem becomes infeasible. Different possible methods (e.g. force filed techniques, etc.) could be developed to reduce the collision probability when the MPC-based method fails to find a feasible solution.

Appendix A

Point Projection on a Line Segment

Given a line segment specified by its two end points as $P_1 = [x_1, y_1]^T \& P_2 = [x_2, y_2]^T$, coordinates of a projected point $P_3 = [x_3, y_3]^T$ named as P_n are calculated by solving the following Equations:

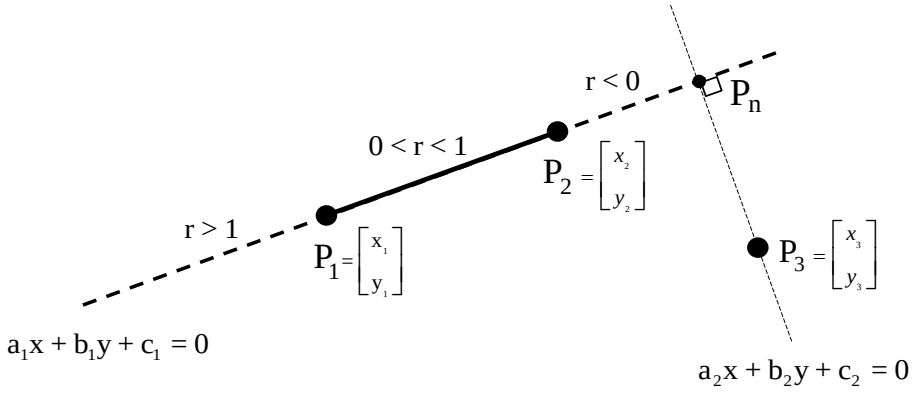

Figure A.1: Point projection on a line segment.

$$a_1 x + b_1 y + c_1 = 0 \tag{A.1}$$

$$a_1 = y_1 - y_2, \quad b_1 = x_2 - x_1$$

$$c_1 = y_2 x_1 - y_1 x_2$$

$$a_2 x + b_2 y + c_2 = 0 \tag{A.2}$$

$$a_2 = x_1 - x_2, \quad b_2 = y_1 - y_2$$

$$c_2 = y_3(y_2 - y_1) + x_3(x_2 - x_1)$$

The solution is :

$$P_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \frac{c_1 b_2 - c_2 b_1}{a_2 b_1 - a_1 b_2} \\ \frac{a_2 c_1 - a_1 c_2}{a_1 b_2 - a_2 b_1} \end{bmatrix} \tag{A.3}$$

Using *r-parameterization* method in Equation (5.12), one can calculated the corresponding "r" as:

$$rP_1 + (1 - r)P_2 = P_n \tag{A.4}$$

$$\Rightarrow$$

$$r = \frac{x_n - x_2}{x_1 - x_2} \tag{A.5}$$

or

$$r = \frac{y_n - y_2}{y_1 - y_2} \tag{A.6}$$

Bibliography

Abbott, J. J., Marayong, P., and Okamura, A. M. (2007). Haptic virtual fixtures for robot-assisted manipulation. In *Robotics research*, pages 49–64. Springer.

Alfano, P. L. and Michel, G. F. (1990). Restricting the field of view: Perceptual and performance effects. *Perceptual and motor skills*, **70**(1), 35–45.

Anderson, R. J. and Spong, M. W. (1992). Asymptotic stability for force reflecting teleoperators with time delay. *The International Journal of Robotics Research*, **11**(2), 135–149.

Arcara, P. and Melchiorri, C. (2002). Control schemes for teleoperation with time delay: A comparative study. *Robotics and Autonomous Systems*, **38**(1), 49–64.

Arthur, K. W. (2000). *Effects of field of view on performance with head-mounted displays*. Ph.D. thesis, University of North Carolina.

Ballard, D. H. (1981). Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, **13**(2), 111–122.

Borges, G. A. and Aldon, M.-J. (2004). Line extraction in 2d range images for mobile robotics. *Journal of Intelligent and Robotic Systems*, **40**(3), 267–297.

Castellanos, J. A., Montiel, J., Neira, J., and Tardós, J. D. (1999). The spmap: A probabilistic framework for simultaneous localization and map building. *Robotics and Automation, IEEE Transactions on*, **15**(5), 948–952.

Cheung, Y., Chung, J. H., and Coleman, N. P. (2009). Semi-autonomous formation control of a single-master multi-slave teleoperation system. In *Computational Intelligence in Control and Automation, 2009. CICA 2009. IEEE Symposium on*, pages 117–124. IEEE.

Diolaiti, N. and Melchiorri, C. (2002). Teleoperation of a mobile robot through haptic feedback. In *Haptic Virtual Environments and Their Applications, IEEE International Workshop 2002 HAVE*, pages 67–72. IEEE.

Droge, G. and Egerstedt, M. (2011). Adaptive look-ahead for robotic navigation in unknown environments. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1134–1139. IEEE.

Duda, R. O. and Hart, P. E. (1972). Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, **15**(1), 11–15.

Falcone, P., Borrelli, F., Asgari, J., Tseng, H. E., and Hrovat, D. (2007). Predictive active steering control for autonomous vehicle systems. *Control Systems Technology, IEEE Transactions on*, **15**(3), 566–580.

Ferre, M., Buss, M., Aracil, R., Melchiorri, C., and Balaguer, C. (2007). *Advances in telerobotics*. Springer Publishing Company, Incorporated.

Ferrell, W. R. and Sheridan, T. B. (1967). Supervisory control of remote manipulation. *Spectrum, IEEE*, **4**(10), 81–88.

Fong, T. W., Conti, F., Grange, S., and Baur, C. (2001). Novel interfaces for remote driving: gesture, haptic, and pda. In *Intelligent Systems and Smart Manufacturing*, pages 300–311. International Society for Optics and Photonics.

Forsberg, J., Larsson, U., Ahman, P., and Wernersson, A. (1993). The hough transform inside the feedback loop of a mobile robot. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 791–798. IEEE.

Forsberg, J., Larsson, U., and Wernersson, A. (1995). Mobile robot navigation using the range-weighted hough transform. *Robotics & Automation Magazine, IEEE*, **2**(1), 18–26.

Glennon, J. S. (1998). *Feature-based Localization In Sonar-Equipped Autonomous Mobile Robots Through Hough Transform and Unsupervised Learning Network*. Master's thesis, Naval Postgraduate School.

Gutmann, J.-S. and Schlegel, C. (1996). Amos: Comparison of scan matching approaches for self-localization in indoor environments. In *Advanced Mobile Robot, 1996., Proceedings of the First Euromicro Workshop on*, pages 61–67. IEEE.

Hashtrudi-Zaad, K. and Salcudean, S. E. (2001). Analysis of control architectures for teleoperation systems with impedance/admittance master and slave manipulators. *The International Journal of Robotics Research*, **20**(6), 419–445.

Hokayem, P. F. and Spong, M. W. (2006). Bilateral teleoperation: An historical survey. *Automatica*, **42**(12), 2035–2057.

Horn, B. K. P. (1986). *Robot Vision: MIT Electrical Engineering and Computer Science SE*. MIT Press.

Hough, P. (1962). Method and means for recognizing complex patterns. US Patent 3,069,654.

Howard, T., Green, C., and Kelly, A. (2010). Receding horizon model-predictive control for mobile robot navigation of intricate paths. In *Field and Service Robotics*, pages 69–78. Springer.

Iyengar, S. and Elfes, A. (1991). *Autonomous Mobile Robots: Perception, mapping, and navigation*. IEEE Computer Society Press.

Jensfelt, P. and Christensen, H. (1998). Laser based position acquisition and tracking in an indoor environment. In *International Symposium on Robotics and Automation-ISRA*, volume 98. Citeseer.

Katsura, S., Suzuyama, T., and Ohishi, K. (2007). A realization of multilateral force feedback control for cooperative motion. *Industrial Electronics, IEEE Transactions on*, **54**(6), 3298–3306.

Keviczky, T., Falcone, P., Borrelli, F., Asgari, J., and Hrovat, D. (2006). Predictive control approach to autonomous vehicle steering. In *American Control Conference, 2006*, pages 6–pp. IEEE.

Kim, H. J., Shim, D. H., and Sastry, S. (2002a). Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles. In *American Control Conference, 2002. Proceedings of the 2002*, volume 5, pages 3576–3581. IEEE.

Kim, K., Lee, H., Park, J., and Yang, M. (2002b). Robotic contamination cleaning system. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 2, pages 1874–1879. IEEE.

Krüsi, P., Pivtoraiko, M., Kelly, A., Howard, T. M., and Siegwart, R. (2010). Path set relaxation for mobile robot navigation. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space*, pages 456–463.

Lapierre, L., Zapata, R., and Lepinay, P. (2007). Combined path-following and obstacle avoidance control of a wheeled robot. *The International Journal of Robotics Research*, **26**(4), 361–375.

Larsson, U., Forsberg, J., and Wernersson, A. (1996). Mobile robot localization: integrating measurements from a time-of-flight laser. *Industrial Electronics, IEEE Transactions on*, **43**(3), 422–431.

Lawrence, D. A. (1993). Stability and transparency in bilateral teleoperation. *Robotics and Automation, IEEE Transactions on*, **9**(5), 624–637.

Lee, D. and Spong, M. W. (2005). Bilateral teleoperation of multiple cooperative robots over delayed communication networks: theory. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 360–365. IEEE.

Lee, D., Martinez-Palafox, O., and Spong, M. W. (2006). Bilateral teleoperation of a wheeled mobile robot over delayed communication network. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 3298–3303. IEEE.

Lee, S., Sukhatme, G., Kim, G. J., and Park, C.-M. (2005). Haptic teleoperation of a mobile robot: a user study. *Presence: Teleoperators & Virtual Environments*, **14**(3), 345–365.

Liu, C., Chen, W.-H., and Andrews, J. (2010). Optimisation based control framework for autonomous vehicles: algorithm and experiment. In *Mechatronics and Automation (ICMA), 2010 International Conference on*, pages 1030–1035. IEEE.

Lumelsky, V. J. and Cheung, E. (1993). Real-time collision avoidance in teleoperated whole-sensitive robot arm manipulators. *Systems, Man and Cybernetics, IEEE Transactions on*, **23**(1), 194–203.

Madhavan, R., Durrant-Whyte, H., and Dissanayake, G. (2002). Natural landmark-based autonomous navigation using curvature scale space. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 4, pages 3936–3941. IEEE.

Mallem, M., Chavand, F., and Colle, E. (1992). Computer-assisted visual perception in teleoperated robotics. *Robotica*, **10**, 93–103.

Malysz, P. (2011). *A Kinematic Control Framework for Asymmetric Semi-autonomous Teleoperation Systems*. Ph.D. thesis, McMaster University.

Malysz, P. and Sirouspour, S. (2011). A kinematic control framework for single-slave asymmetric teleoperation systems. *Robotics, IEEE Transactions on*, **27**(5), 901–917.

Malysz, P. and Sirouspour, S. (2013). Mixed autonomous/teleoperation control of asymmetric robotic systems. *Submitted for publication to IEEE Transactions on Automatic Control*.

Nanayakkara, D., Kiguchi, K., Murakami, T., Watanabe, K., and Izumi, K. (2001).

Skillful adaptation of a 7-dof manipulator to avoid moving obstacles in a teleoperated force control task. In *Industrial Electronics, 2001. Proceedings. ISIE 2001. IEEE International Symposium on*, volume 3, pages 1982–1987. IEEE.

Nguyen, V., Gächter, S., Martinelli, A., Tomatis, N., and Siegwart, R. (2007). A comparison of line extraction algorithms using 2d range data for indoor mobile robotics. *Autonomous Robots*, **23**(2), 97–111.

Nielsen, C. W., Goodrich, M. A., and Ricks, R. W. (2007). Ecological interfaces for improving mobile robot teleoperation. *Robotics, IEEE Transactions on*, **23**(5), 927–941.

Niemeyer, G. and Slotine, J.-J. E. (2004). Telemanipulation with time delays. *The International Journal of Robotics Research*, **23**(9), 873–890.

Nourbakhsh, I. R., Sycara, K., Koes, M., Yong, M., Lewis, M., and Burion, S. (2005). Human-robot teaming for search and rescue. *Pervasive Computing, IEEE*, **4**(1), 72–79.

Park, J., Kim, D., Yoon, Y., Kim, H., and Yi, K. (2009). Obstacle avoidance of autonomous vehicles based on model predictive control. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, **223**(12), 1499–1516.

Pfister, S. T., Roumeliotis, S. I., and Burdick, J. W. (2003). Weighted line fitting algorithms for mobile robot map building and efficient data representation. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 1, pages 1304–1311. IEEE.

Ponce, J. and Forsyth, D. A. (2002). *Computer Vision: A Modern Approach*. Prentice Hall.

Poncela, A., Urdiales, C., Pérez, E. J., and Sandoval, F. (2009). A new efficiency-weighted strategy for continuous human/robot cooperation in navigation. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, **39**(3), 486–500.

Premebida, C. and Nunes, U. (2005). Segmentation and geometric primitives extraction from 2d laser range data for mobile robot applications. *Robotica*, pages 17–25.

Rubi, J., Rubio, A., and Avello, A. (2002). Involving the operator in a singularity avoidance strategy for a redundant slave manipulator in a teleoperated application. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2973–2978. IEEE.

Salcudean, S. E., Zhu, M., Zhu, W.-H., and Hashtrudi-Zaad, K. (2000). Transparent bilateral teleoperation under position and rate control. *The International Journal of Robotics Research*, **19**(12), 1185–1202.

Sanders, D. (2010). Comparing ability to complete simple tele-operated rescue or maintenance mobile-robot tasks with and without a sensor system. *Sensor Review*, **30**(1), 40–50.

Schenker, P. S., Huntsberger, T. L., Pirjanian, P., Baumgartner, E. T., and Tunstel, E. (2003). Planetary rover developments supporting mars exploration, sample return and future human-robotic colonization. *Autonomous Robots*, **14**(2-3), 103–126.

Sheridan, T. B. (1989). Telerobotics. *Automatica*, **25**(4), 487–507.

Sheridan, T. B. (1992). Musings on telepresence and virtual presence. *Presence: Teleoperators and virtual environments*, **1**(1), 120–126.

Sirouspour, S. (2005). Modeling and control of cooperative teleoperation systems. *Robotics, IEEE Transactions on*, **21**(6), 1220–1225.

Sirouspour, S. and Shahdi, A. (2005). Bilateral teleoperation under communication time delay using an lqg controller. In *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, pages 1257–1262. IEEE.

Slawiñski, E., Mut, V., Salinas, L., and García, S. (2012). Teleoperation of a mobile robot with time-varying delay and force feedback. *Robotica*, **30**(01), 67–77.

Thrun, S., Burgard, W., and Fox, D. (1998). A probabilistic approach to concurrent mapping and localization for mobilerobots. *Machine learning*, **31**(1-3), 29–53.

Tzafestas, C. S. (2007). Virtual and mixed reality in telerobotics: A survey. *Industrial Robotics: Programming, Simulation and Applications*, pages 437–470.

Vandorpe, J., Van Brussel, H., and Xu, H. (1996). Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2d range finder. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 1, pages 901–908. IEEE.

Werling, M. and Liccardo, D. (2012). Automatic collision avoidance using model-predictive online optimization. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pages 6309–6314. IEEE.

Yoerger, D. R. (1982). Supervisory control of underwater telemanipulators: Design and experiment. Technical report, DTIC Document.

Yokokohji, Y. and Yoshikawa, T. (1994). Bilateral control of master-slave manipulators for ideal kinesthetic coupling-formulation and experiment. *Robotics and Automation, IEEE Transactions on*, **10**(5), 605–620.

Yokokohji, Y., Imaida, T., and Yoshikawa, T. (2000). Bilateral control with energy balance monitoring under time-varying communication delay. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 3, pages 2684–2689. IEEE.

Yoon, Y., Shin, J., Kim, H. J., Park, Y., and Sastry, S. (2009). Model-predictive active steering and obstacle avoidance for autonomous ground vehicles. *Control Engineering Practice*, **17**(7), 741–750.

Zhang, L. and Ghosh, B. K. (2000). Line segment based map building and localization using 2d laser rangefinder. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 3, pages 2538–2543. IEEE.