

**AUGMENTED REALITY-BASED
MODEL-MEDIATED TELEOPERATION: A
MOBILE TELEROBOT CASE STUDY**

**A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of**

MASTER OF SCIENCE

in Mechanical Engineering

**by
Nihat Çağhan KİRİŞCİ**

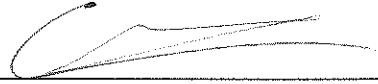
**July 2019
İZMİR**

We approve the thesis of Nihat Çağhan KIRIŞCI

Examining Committee Members:



Prof. Dr. Enver TATLICIOĞLU
Department of Electrical & Electronics Engineering
İzmir Institute of Technology



Assoc. Prof. Dr. Mehmet İsmet Can DEDE
Department of Mechanical Engineering
İzmir Institute of Technology

Assist. Prof. Dr. Özgün BAŞER
Department of Mechatronics Engineering
İzmir Kâtip Çelebi University

18 July 2019



Assoc. Prof. Dr. Mehmet İsmet Can DEDE
Supervisor
Department of Mechanical Engineering
İzmir Institute of Technology



Prof. Dr. Sedat AKKURT
Head of the Department of
Mechanical Engineering

Prof. Dr. Aysun SOFUOĞLU
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my dear family, especially my parents Ayşe and Ergün Kirişçi for their endless support, patience, encouragements, and understanding when I was not able to spend time with them during my studies.

I am sincerely grateful to my supervisor Dr. Mehmet İsmet Can Dede for his patience, guidance and support. He allowed this study to be my own work, but guided me in the right direction when needed.

I would like to thank the rest of my thesis committee: Prof. Enver Tatlıcıoğlu and Dr. Özgün Başer for their insightful comments and suggestions.

I am indebted to Dr. Emre Uzunoğlu and Gizem Ateş without whom it would not be possible to complete my thesis. Their moral and technical support is greatly appreciated.

I wish to thank the members of IRL, RAML, Modeling & Prototyping Lab, Human Robot Interaction Lab, in no particular order, for their contribution to this project, keeping my morale up, and simply being my friends: Osman Nuri Şahin, Omar Maarooof, Barış Taner, Murat Demirel, Cevahir Karagöz, Orhan Ayit, Mert Kanık, Elvan Doğan Kumtepe, İbrahimcan Görgülü, Emir Mobedi, Oğulcan Işıtman, Barış Çelik, Furkan Küçüköğlü, Egecan Türk, Erkan Paksoy, Hazal Emet, Uğur Nalbant, Şerif Nadir Soyaker, and those who I may have forgotten.

In addition, I would like to thank my friends Gizem Ersavaş and Merve Özkahya for sticking together during our thesis studies including many sleepless nights.

Lastly, I acknowledge the contribution of my friends, Emir "CoolKat" Cilvez, Oğulcan "tötte" Sandıkçı, Soykan "Bolin" Ataman, and Özer "Debbag" Debbag without whom this thesis would have been completed months earlier.

This study is funded by *The Scientific and Technological Research Council of Turkey (TÜBİTAK)* with the project name *Robotic Squid for Underwater Manipulation and Intervention* and grant number 216M219.

ABSTRACT

AUGMENTED REALITY-BASED MODEL-MEDIATED TELEOPERATION: A MOBILE TELEROBOT CASE STUDY

Teleoperation is defined as operating a robot in a remote environment. Teleoperation utilizes the strength and dexterity of robots and the interpretation and problem solving skills of humans. In a teleoperation system, the robot to be controlled is referred as the slave. The master is a device that the human operator interacts with to send commands to the slave or receive feedback from the slave environment such as haptic or audio.

However, teleoperation of a robot in an unknown environment solely based on haptic and visual feedback is a demanding task. The effects of time delay in communication channels makes completing this task even more challenging. Model-Mediated Teleoperation (MMT) aims to solve this problem by creating a virtual model of the slave robot and the environment. This virtual model receives commands from the master and returns haptic feedback just as the real slave robot is interacting with the environment, effectively with no delay. However, without actually knowing where the position of the virtual robot corresponds in the real environment, it is still challenging to carry out the task. In this project, a novel augmented reality based method is proposed to render the virtual robot into the real life live video feed. Integration of the non-delayed robot into the real environment intends to solve this problem by enhancing the perception of the user.

ÖZET

ARTIRILMIŞ GERÇEKLİK TEMELLİ MODEL ARACILI TELEOPERASYON: BİR MOBİL TELEROBOT ÇALIŞMASI

Teleoperasyon, uzak ortamdaki bir robotu kontrol etmek olarak tanımlanır. Teleoperasyonda, robotların gücünden ve manevra kabiliyetlerinden, insanların ise bir durumu algılama ve çözüm üretme yeteneklerinden faydalanılır. Bir teleoperasyon sisteminde, kontrol edilen robot, bağımlı robot olarak adlandırılır. İnsan kullanıcının etkileşimde bulunarak bağımlı sisteme sinyaller gönderdiği ve bağımlı robotun bulunduğu ortamın bilgisini haptik veya duysal geribildirim olarak aldığı bileşene ise ana/ana sistem denir.

Bilinmeyen bir ortamda bulunan robotun, yalnızca haptik veya görsel geribildirim ile kontrol edilmesi zor bir iştir. Ana ile bağımlı sistemler arasındaki haberleşme kanallarından kaynaklanan gecikme ve paket kaybı gibi sorunlar bu işlemi daha da zorlaştırır. Model aracılı teleoperasyon yöntemi, uzak ortamın sanal bir modelini oluşturarak bu sorunun üstesinden gelmeyi amaçlar. Bu sanal model, kullanıcıdan komutlar alarak, hiç gecikme yokmuş gibi gerçek ortamın vereceği tepkiyi taklit eder. Ancak, sanal robotun gerçek ortamda nereye denk geldiğini bilmeden işlemi gerçekleştirmek hala zordur. Bu projede artırılmış gerçeklik bazlı özgün bir teleoperasyon methodu sunulmuştur. Sanal robot, gerçek ortamdaki gelen görüntünün üzerine işlenerek kullanıcının algısını, dolayısıyla teleoperasyonun performansını artırmak hedeflenmiştir.

I would like to dedicate this thesis to my dear parents Ayşe
and Ergün KİRİŞCİ.

TABLE OF CONTENTS

LIST OF FIGURES	ix
LIST OF TABLES	xi
CHAPTER 1. INTRODUCTION	1
1.1. Control Methods for Compensation in Communication Failures ...	4
1.2. Model-mediated Teleoperation	8
1.3. Aim of the Thesis	11
1.4. Contributions	12
1.5. Outline	12
CHAPTER 2. BACKGROUND ON MODEL-MEDIATED TELEOPERATION ...	13
CHAPTER 3. METHODOLOGY	17
3.1. Proposed Model Mediated Teleoperation Method	17
3.2. Experimental Setup	20
3.2.1. Master Device	21
3.2.2. Slave Device	22
3.2.2.1. Hardware	23
3.2.2.2. Kinematics	24
3.2.3. Workspace Requirements	26
3.3. Control System Implementation	27
3.4. Augmented Reality	30
CHAPTER 4. EXPERIMENTAL TESTS	34
4.1. Augmented Reality Tests	34
4.2. Teleoperation Performance Tests	35
4.3. Teleoperation Performance Test Results	39
4.3.1. Controller Performance Results	39
4.3.2. User Test Results	43
4.3.3. User Questionnaire Results	49
CHAPTER 5. CONCLUSION	51

REFERENCES	53
APPENDICES	
APPENDIX A. TEST DATA	56

LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
Figure 1.1. Teleoperation illustration.	1
Figure 1.2. Unilateral teleoperation.	2
Figure 1.3. Bilateral teleoperation.	2
Figure 1.4. Conceptual design of NASA’s Restore-L.	3
Figure 1.5. Aselsan’s Kaplan, unmanned ground vehicle.	4
Figure 1.6. H2000-SUR remotely operated vehicle.	4
Figure 1.7. Da Vinci surgical teleoperation system.	5
Figure 1.8. Illustration of earlier control methods for teleoperations subjected to time delays.	6
Figure 1.9. Bilateral teleoperation scheme with wave variables.	7
Figure 1.10. Wave transformation between incoming and outgoing signals.	7
Figure 1.11. Model-mediated teleoperation.	9
Figure 1.12. Representation of the master, the proxy and the slave in one DoF proof of concept.	10
Figure 1.13. Force generation through the master and the proxy.	10
Figure 1.14. Model update procedure if the virtual surface is below the estimated surface.	11
Figure 2.1. Grid based virtual model.	14
Figure 3.1. Proposed teleoperation scheme.	18
Figure 3.2. Critical zone for haptic rendering.	19
Figure 3.3. An example generated map.	20
Figure 3.4. Overview of the test setup.	21
Figure 3.5. Novint Falcon desktop haptic device.	22
Figure 3.6. The mobile robot.	23
Figure 3.7. Geometrical representation of the robot.	25
Figure 3.8. ROS node locations	27
Figure 3.9. Illustration of ROS nodes running on Odroid.	28
Figure 3.10. High level position and low level velocity controllers.	28
Figure 3.11. Illustration of ROS nodes running on the master PC.	29
Figure 3.12. The 3D model of the slave robot.	30
Figure 3.13. The pinhole camera model.	31
Figure 4.1. Uncalibrated augmented vision.	34

Figure 4.2. Calibrated augmented vision.	35
Figure 4.3. Bilateral teleoperation experiment setup.	36
Figure 4.4. Bilateral teleoperation experiment setup.	37
Figure 4.5. Virtual reality based model mediated teleoperation test setup.	38
Figure 4.6. Augmented reality based model mediated teleoperation test setup.	38
Figure 4.7. Proxy and robot positions on x-axis of the world frame	40
Figure 4.8. Matched proxy and robot positions on x-axis of the world frame	40
Figure 4.9. Proxy and robot positions on y-axis of the world frame	41
Figure 4.10. Matched proxy and robot positions on y-axis of the world frame	41
Figure 4.11. Position tracking error in x-axis of the world frame	42
Figure 4.12. Position tracking error in y-axis of the world frame	42
Figure 4.13. Proxy and robot trajectories for teleoperation test.	45
Figure 4.14. Proxy and robot trajectories for VR test.	45
Figure 4.15. Proxy and robot trajectories for AR test.	47
Figure 4.16. Mean position of the robot while completing the checkpoints during standard teleoperation test.	47
Figure 4.17. Mean position of the robot while completing the checkpoints during VR test.	48
Figure 4.18. Mean position of the robot while completing the checkpoints during AR test.	48
Figure 4.19. Mean position of the robot while completing the checkpoints during all tests.	49

LIST OF TABLES

<u>Table</u>	<u>Page</u>
Table 3.1. Technical specifications of Novint Falcon haptic device.	22
Table 4.1. Position tracking errors in x/y-axes and in total.	43
Table 4.2. Completion times for each subject and test	43
Table 4.3. User questionnaire results (1: No method, 2: VR method, 3: AR method).	50

CHAPTER 1

INTRODUCTION

Teleoperation can be defined as extending one's abilities to a remote environment. This is achieved by cooperation of humans and robots as they both excel at different abilities. Teleoperation builds a bridge between these abilities: control, perception, and planning of humans; accuracy, resilience, and mobility of robots (Sheridan, 1989). The need for teleoperation arises whenever either of the previously mentioned parties are not able to perform the operation by themselves. teleoperation is utilized when the environment of the task is difficult for a human to reach such as underwater and space operations, or where it poses a threat to human health such as nuclear reactors or potential explosions (Trevelyan et al., 2008).

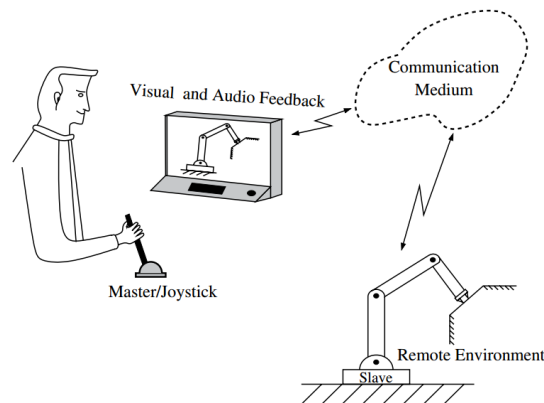


Figure 1.1. Teleoperation illustration.
(Source: Hokayem and Spong 2006)

A robot teleoperation usually contains three main elements as illustrated in Figure 1.1. A master control system directed by a human user, a slave robot controlled by the master and a communication line providing exchange of data between the master and the slave. Position or velocity data captured from the operator is transmitted to the slave. In return, master receives haptic and/or visual feedback. Teleoperation can be categorized according to data flow between the master and the slave.

Illustrated in Figure 1.2, in unilateral teleoperation, information is transmitted in one way only, from the master to the slave. Environment conditions are not captured by the slave and it is limited to visual feedback only. However, in bilateral teleoperation,



Figure 1.2. Unilateral teleoperation.

additional feedback such as audio or haptics is transferred to the master side as shown in Figure 1.3.

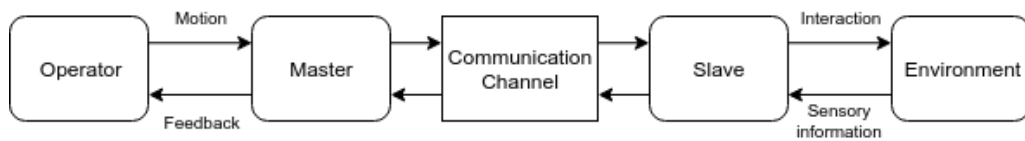


Figure 1.3. Bilateral teleoperation.

Usually, motion (velocity/position) or force signals are exchanged between master and slave systems. Depending on the number of the communication channels, a bilateral teleoperation can either be two or four channels. In two channel systems, one signal is transmitted from the master to the slave and one is received in return, resulting in a total of two channels. On the other hand, four channel systems intend to enhance the telepresence of the user by sending and receiving both motion and force signals. *Telepresence* is the feeling of being physically present at the slave environment as if the task is getting carried out by the user without the median slave (Stassen and Smets, 1995). It is achieved by gathering sufficient information about the slave and the environment as well as transmitting it to the operator in a natural way (Sheridan, 1989).

Teleoperation can also be divided into two categories depending on the workspace of the slave robot. Limited-workspace teleoperation implies that the slave can be operated in a bounded area. A stationary manipulator is used as slave and the end effector usually mimics the hand movement of the human operator such as gripping objects and manipulating the environment. When the slave is a mobile robot such as a remotely operated vehicle, the workspace becomes unlimited. Therefore the teleoperation is called unlimited-workspace. Unlimited-workspace teleoperation is mainly used to explore or inspect dangerous environments.

Teleoperation is a widely used method in today's robotics field. There are many reasons why teleoperation is preferred over other methods in control of robots. The main reason would be to make use of human judgment and decision-making while prioritizing

safety, combined with the strength and dexterity of robots (Stakem, 2014). The applications of teleoperation mainly consist of operating under hazardous and/or hard-to-reach environments such as handling nuclear materials, inspecting bomb sites, and performing maintenance in space.



Figure 1.4. Conceptual design of NASA's Restore-L.
(Source: NASA)

Teleoperation is vital in maintenance operations in space. Sending humans to space to perform a simple operation is both immensely costly and risky compared to carrying out the task using a robot. An example would be NASA's Restore-L (Fig. 1.4) which is an ongoing project to repair, refuel, and even relocate satellites to prolong their lifespans. In addition, space shuttles of NASA are equipped with robotic arms to provide self-maintenance.

Military is another field that takes advantage of teleoperation technology. The robots are used for surveying, neutralizing explosives, clearing routes, and locating targets while preserving the lives of the soldiers. Figure 1.5 shows Kaplan, unmanned ground vehicle (UGV), manufactured by Aselsan. It can operate autonomously or be operated by teleoperation where human intervention is required.

UGVs are not the only unlimited-workspace robots that are used in teleoperation. Unmanned underwater vehicles have one of the largest shares in mobile robot industry. The main uses are exploration, surveying, and security.

The precision, endurance, and durability of the robots are also utilized in medical field. Surgical robots are getting more popular and used every day. Figure 1.7 shows



Figure 1.5. Aselsan's Kaplan, unmanned ground vehicle.
(Source: Aselsan)



Figure 1.6. H2000-SUR remotely operated vehicle.
(Source: Eca)

Da Vinci surgical teleoperation system, one of the world's best-known and most used surgical robots. The slave robot has four serial arms. These manipulators are controlled by the master system that has two grippers, foot pedals and a stereoscopic vision display. Since robotic arms enter the surgery area through small incisions, the healing period is minimal. The system also allows the surgeon to sit during the operation which greatly helps with the endurance.

1.1. Control Methods for Compensation in Communication Failures

The type of a teleoperation is determined based on certain variables. The application that teleoperation is implemented on, conditions of the slave environment, and issues



Figure 1.7. Da Vinci surgical teleoperation system.
(Source: DaVinci)

in communication channel are the main factors in deciding the teleoperation method. When there is negligible or no delay in communication between the master and the slave, previously discussed direct bilateral teleoperation method is sufficient to reflect the sense of touch to the user and achieve telepresence.

Even small time delays such as 10-100 ms affect the stability and the transparency of the entire system greatly (Lawrence, 1993). An obvious workaround to this problem is to use the unilateral teleoperation method and rely on delayed visual feedback only. As Ferrell (1965) demonstrated, the users tend to develop a *move and wait* strategy where the user makes a move and waits until the slave stops. However, in this method, the task completion time is dependent on the time delay and the telepresence is lost.

Instead of directly controlling every move of the slave, a supervisory method is proposed by Ferrell and Sheridan (1967). In this method, the slave has increased autonomy and able to perform predefined tasks requested by the master. The task completion time of this method does not depend on the time delay but on the slave's abilities. Additional control loops are included in supervisory control. The remote loop is responsible for receiving the commands from the master side and the automation of the slave. The supervisory and local loops are between the master system and the operator. In the supervisory loop, the operator utilizes visual feedback and sets short time goals for the slave where as the local loop contains a model of the slave and may provide *quasi-static feed-*

back. Representations of these methods are given in Figure 1.8.

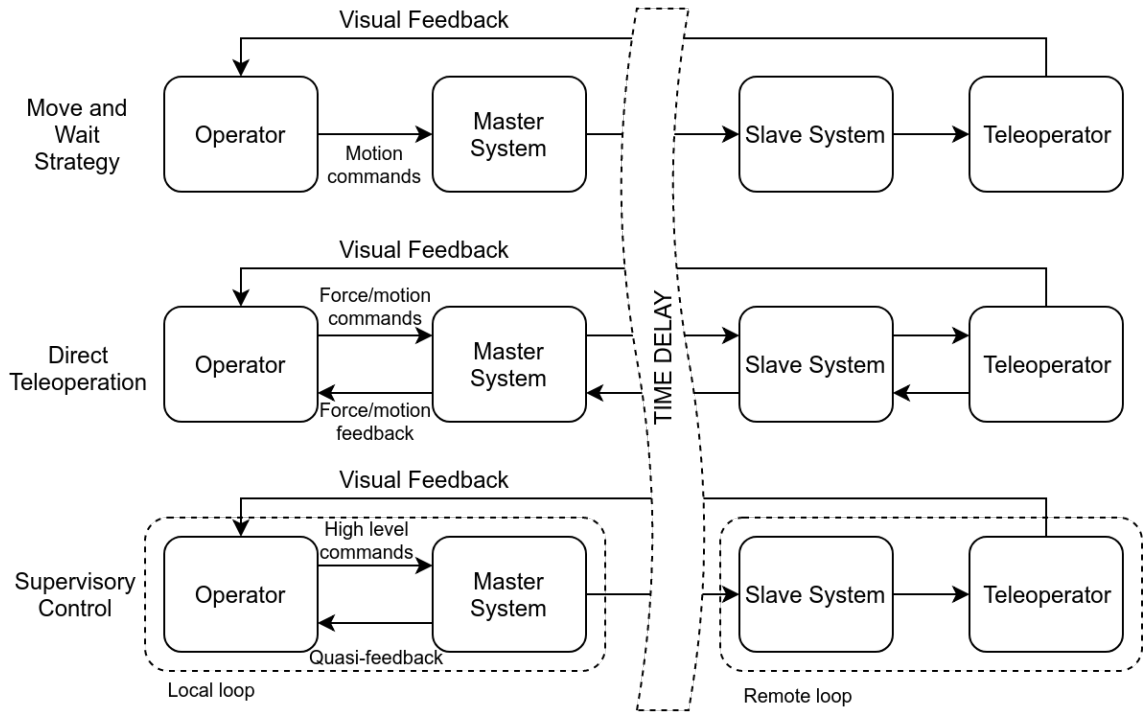


Figure 1.8. Illustration of earlier control methods for teleoperations subjected to time delays.

Anderson and Spong (1989) developed a leading control strategy based on scattering theory in which the stability was maintained if the *passivity* of the system is ensured, and allowed the system to have a power scaling without loss of stability. A system is said to be *passive* if more energy is absorbed than produced. In other words, the power input to the system should be stored or dissipated and the system cannot provide more energy than initially stored. In Equation 1.1, P_{in} is the power input to the system, where u is the input vector and y is the output vector.

$$P_{in} = u^T y \quad (1.1)$$

An *energy store* function, E_{store} , that is lower bounded by the constant E_{min} is defined. Energy dissipation in the system is represented by the non-negative power dissipation function, P_{diss} . If E_{min} is taken as zero, passivity condition is becomes:

$$P_{in} = \frac{d}{dt} E_{store} + P_{diss} \quad (1.2)$$

The passivity condition can also be written in the integral form to be used in discrete calculations:

$$\int_0^t P_{in} d\tau = E_{store}(t) - E_{store}(0) + \int_0^t P_{diss} \geq -E_{store}(0) \quad \forall t \geq 0 \quad (1.3)$$

The system is said to be *lossless* if P_{diss} term is always zero and *dissipative* if it is positive.

Another passivity based method was introduced by Niemeyer (1996), taking advantage of wave variables in bilateral teleoperation with haptic feedback. Wave transformations are applied at both master and slave sides, marked as **b** in Figure 1.9.

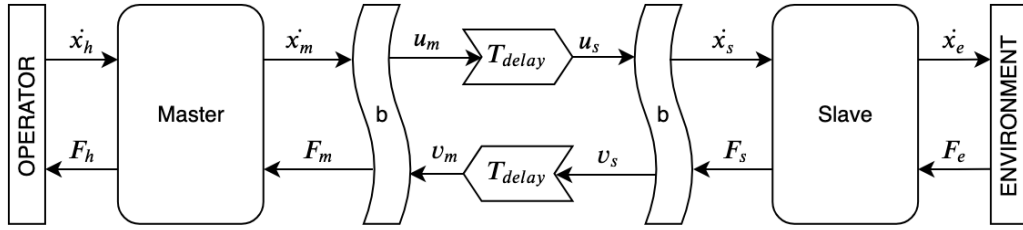


Figure 1.9. Bilateral teleoperation scheme with wave variables.

As shown in Figure 1.10, wave transformation applies a conversion between incoming (\dot{x}, v) and outgoing (F, u) signals where u is referred as forward/right moving and v as backward/left moving wave.

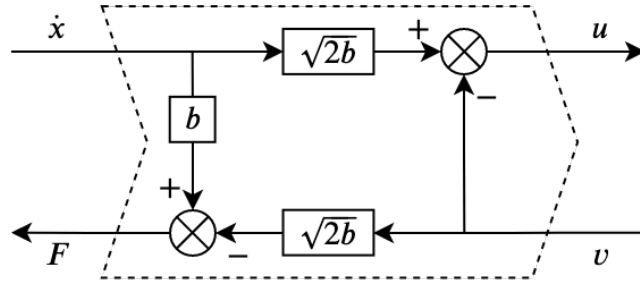


Figure 1.10. Wave transformation between incoming and outgoing signals.

The transformation of the waves can be calculated from the Equation 1.4.

$$u = \frac{b\dot{x} + F}{\sqrt{2b}}, \quad v = \frac{b\dot{x} - F}{\sqrt{2b}} \quad (1.4)$$

where b is a positive scalar or a symmetric positive definite matrix depending on the DoF of the system.

The reverse transformation can easily be deduced from the Equation 1.4 by taking the sum and the difference of the wave variables u and v .

$$\dot{x} = \sqrt{\frac{1}{2b}}(u + v), \quad F = \sqrt{\frac{b}{2}}(u - v) \quad (1.5)$$

Once the power input is expressed in terms of wave variables(Equation 1.6), the passivity condition can be rewritten as well (Equation 1.7).

$$P_{in} = \dot{x}^T F = \frac{1}{2}u^T u - \frac{1}{2}v^T v \quad (1.6)$$

$$\int_0^t \frac{1}{2}v^T v d\tau \leq \int_0^t \frac{1}{2}u^T u d\tau + E_{stored}(0) \quad \forall t \geq 0 \quad (1.7)$$

The teleoperation control methods discussed in this section each have distinct use cases and properties. Move-and-wait strategy is employed in systems with relatively larger communication time delays. Similar to move-and-wait strategy, supervisory control is also employed when there are relatively larger time delays. In both of these methods, force feedback is not fed back to the user. However, in supervisory control, the slave system has its own autonomy that awaits the user's input via the communication line. Therefore, supervisory control is highly dependent on the autonomy level of the slave robot and the user is not in direct control of the robot. The other methods mentioned in this section are employed for direct teleoperation. Wave variables method guarantees the stability of a system subjected to time delays in communication while compromising the transparency of the system. Considering the disadvantages in the aforementioned methods, model-mediated teleoperation method aims to keep both the transparency and the telepresence high while suppressing the negative effects of time delay.

Model-mediated teleoperation (MMT) method was proposed by Mitra and Niemeyer (2008b). MMT aims to enhance the performance of the teleoperation by providing telepresence to the user even under communication delays. The following chapter explains this method in detail.

1.2. Model-mediated Teleoperation

Telepresence is the main reason bilateral teleoperation is preferred over unilateral. The sense of the user being in and interacting with the environment greatly enhances the performance of the teleoperation. Time delay in bilateral teleoperation removes the telepresence in a standard application. MMT method aims to keep the bilateral teleoperation as an applicable solution. The performance of the MMT highly depends on the accuracy of the virtual model and the tracking capabilities of the slave. The accuracy of the model

can be improved by using force sensors to model the stiffness or pre-generating the slave environment by utilizing visual or distance sensors. Also, the visual feedback can be improved by adding more detailed information of the virtual environment (Uzunoğlu and Dede, 2017).

The feedback in bilateral teleoperation introduces the instability problem when the system is subjected to communication failures. Loss or delay in the signals between the master reduces the telepresence of the user. Model-mediated teleoperation method intends to solve this problem by creating a virtual copy of the slave environment. The user interacts with this virtual environment and receives locally rendered haptic feedback. Then, the motion of the master is passed on to the slave side through the delayed communication line where a slave controller controls the slave device.

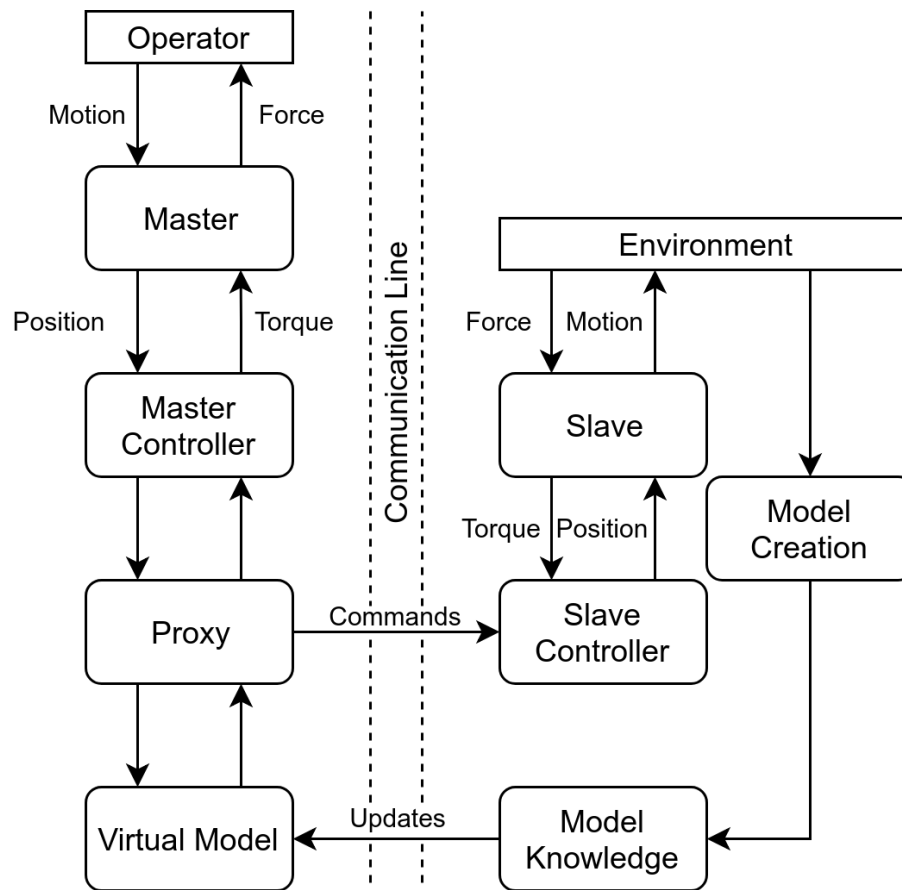


Figure 1.11. Model-mediated teleoperation.

Visualized in Figure 1.11, the left side of the communication line represents the master system. As the slave moves or interacts with the environment, the virtual model is built in the master side from the updates sent from the slave side. *the proxy* is the virtual representation of the slave that resides inside the model. The proxy follows the motion of

the master with its own dynamics that is given in Equation 1.8. Once the proxy catches the master, $\bar{x}_p - \bar{x}_m$ becomes zero. Therefore, the proxy's velocity is equal to the master's velocity. Eventhough the proxy always tries to follows the master, it is not allowed to penetrate into the virtual surface. If the master moves into the surface, the proxy hovers on the surface while keeping its distance to the master at minimum.

$$\dot{\bar{x}}_p = \dot{\bar{x}}_m + \hat{\lambda}(\bar{x}_m - \bar{x}_p) \quad (1.8)$$

where $\hat{\lambda}$ is a diagonal time constant matrix for first-order dynamics, $\dot{\bar{x}}_p$, $\dot{\bar{x}}_m$, \bar{x}_p , and \bar{x}_m , are the velocity and the position column vectors of the proxy and the master, respectively.

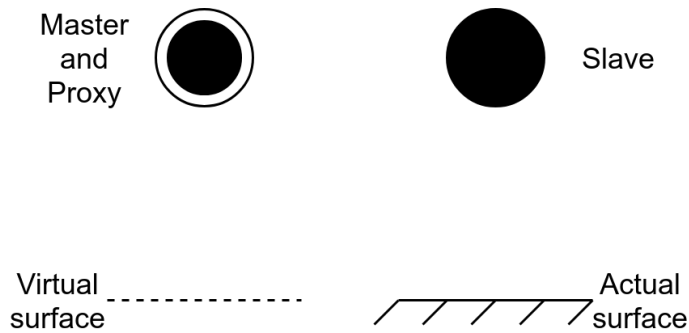


Figure 1.12. Representation of the master, the proxy and the slave in one DoF proof of concept.

$$\bar{F}_m = \hat{K}_{pm}(\bar{x}_p - \bar{x}_m) + \hat{K}_{vm}(\dot{\bar{x}}_p - \dot{\bar{x}}_m) \quad (1.9)$$

\bar{F}_m , the force reflected to the operator through the master device is calculated with PD control gain diagonal matrices of \hat{K}_{pm} and \hat{K}_{vm} , shown in Equation 1.9 and illustrated in Figure 1.13.

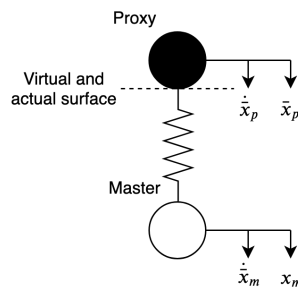


Figure 1.13. Force generation through the master and the proxy.

If a new surface is detected above the master, the virtual surface is placed just below the master to prevent sudden force generation or increase the total energy of the system. As the master moves towards the recently detected surface, the virtual surface is pulled with the master without generating any force (Figure 1.14). However, the virtual surface cannot be moved in the other direction. If the master moves in the other direction, the proxy stays on the surface as described before. When the virtual surface reaches the position of the estimated surface, the interaction returns to normal and the environmental forces are generated again with the Equation 1.9.

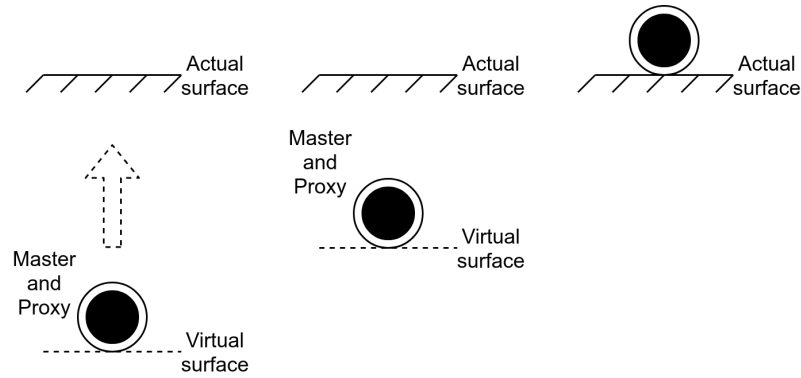


Figure 1.14. Model update procedure if the virtual surface is below the estimated surface.

While the operator is interacting with the virtual environment on the master side, the commands from the master are also transmitted to the slave side. The slave controller makes sure that the slave follows the motion and force demands of the operator. In a force-controlled slave, a PD controller is used to generate force command, F_{pd} (Figure 1.10). The equations for generating force on the master and the slave are quite similar, though \hat{K}_{ps} and \hat{K}_{vs} matrices must be chosen according to the slave dynamics.

$$\bar{F}_{pd} = \hat{K}_{ps}(\bar{x}_p - \bar{x}_s) + \hat{K}_{vs}(\dot{\bar{x}}_p - \dot{\bar{x}}_s) \quad (1.10)$$

1.3. Aim of the Thesis

The aim of this thesis is to extend the existing model mediated teleoperation method to enhance the telepresence of the operator. An augmented reality based vision system is developed, integrated, and tested to eliminate the effects of time delay on the

user during teleoperation. The study consists of building a mobile slave robot including high level position and low level motor controllers, adapting and implementing the model mediated teleoperation method to unlimited workspace, improving the delayed visual feedback received from the slave environment with augmented reality, designing and carrying out experiments to validate the proposed methodology.

1.4. Contributions

This study contributes to the literature by merging two rapidly growing research fields: model-mediated teleoperation and augmented reality. A technology that is strictly aimed towards enhancing the real-life environments with extra perceptual information is integrated into MMT method to provide improved visual feedback to the user.

1.5. Outline

This study is presented in five chapters. In this chapter, the purpose of the thesis is stated after the background information on the relevant studies is given. In the following chapter, previous researches specifically related to model-mediated teleoperation and augmented reality applications are investigated. In chapter 3, all theoretical information about methods, software, and experimental setup used to implement the design are explained in detail. Chapter 4 defines the test procedure and presents the results to evaluate the validity of the proposed method. The thesis is concluded with the Chapter 5, discussing the outcome of the thesis and possible future works for improvement.

CHAPTER 2

BACKGROUND ON MODEL-MEDIATED TELEOPERATION

Model-mediated teleoperation method was first proposed by Mitra and Niemeyer (2008b) as a PhD thesis. The study was a branching point for all future researches in the subject of bilateral teleoperation with time delays. The validation of the method was carried out in a one degree-of-freedom system as a proof of concept and to reduce the computational power required to run the system in real time. It is stated that the performance of this method greatly depends on the accuracy of the virtual model.

- The following research by Mitra and Niemeyer (2008a) focuses on updating the model with user suggestions based on the fact the human is a vital part of the control loop in bilateral teleoperation. The method aims to utilize the adaptive estimation ability of human brain. By inspecting delayed video feedback, the user can pause the teleoperation and update the model through the master interface. Estimated potential contact locations are then transmitted to the slave. If the slave approves the updates by its own sensors, the suggestion of the user is accepted and the teleoperation continues from the pause state.

The test setup consists of a three DoF haptic device and a virtual slave environment. Virtual environment is preferred over a real scenario to have consistency between subjects and methods, to have optimal conditions for different teleoperation methods, and to provide better visual feedback. A toggle switch is placed in virtual environment for users to interact with. The switch has three states: up, down, and broken. The switch can be moved with sufficient force but breaks if the threshold is exceeded. The user is asked to toggle the switch to down state without breaking. The completion time, success rate depending on whether the switch has been broken or not, and user preference. It is found that the user suggested method is preferred over the traditional MMT approach. The new method appears not to have improved the accuracy of the teleoperation. However, user suggestions reduced the time of completion of the task. Future works of this study states the need for extending MMT to multiple degrees of freedom.

- Uzunoğlu and Dede (2017) extended the one DoF experiments by Mitra and Niemeyer

(2008b) to 3 DoF case. Using 3 DoF haptic devices as the master and the slave, the reliability of the method is investigated. Virtual model of the system is created in a grid based environment shown in Figure 2.1.

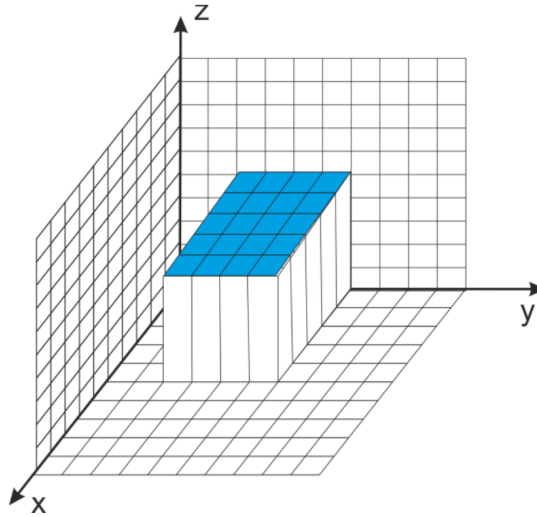


Figure 2.1. Grid based virtual model.

As an addition, an impedance controller was implemented on the slave side to prevent applying undesirable forces to the environment and to ensure the stability of the system.

- Willaert et al. (2012) focused on improving the accuracy of the virtual model and introducing new methods to estimate the geometrical shape of the environment before the contact occurs. Using force sensors and a stereo camera on a 2 DoF system, the position and the angle of the contact were estimated.
- MMT was extended to 6 DoF by Xu et al. (2013). Since building the model for 6 DoF case would be quite challenging with the previous estimation methods, a point-cloud based algorithm was developed using a time-of-flight camera. Point-cloud data captured by the camera is used to generate forces in the virtual environment. Instead of estimating the shape of the surface, haptic feedback is directly produced from the points. A surface normal is calculated considering the points only within a specific distance from the master.
- Another difficulty in MMT is model jumps. When the slave touches an object, it is not instantly updated in the model due to time delay. This situation may cause

step force generation on the master side. Previous solutions on this problem compromised the transparency of the system which is against the governing idea of the MMT: telepresence. Yazdankhoo and Beigzadeh (2019) proposed a method that interrupts the teleoperation whenever the model is updated. Until the stability of the system is gained again, both the master and the slave are controlled by their own independent controllers. The method is able to prevent the jumps regardless of the environment type. However, interrupting the connection to the slave may be inconvenient to the operator.

- The studies on MMT were conducted on single-user systems so far. MMT on a multi-master multi-slave system were implemented by Passenberg et al. (2010) utilizing admittance controllers on the master sides. This technique allowed the cooperation of more than one master-slave subsystem to manipulate the same object.
- MMT method was invented to be used in limited workspace operations. Rapidly growing mobile robot scene demanded new teleoperation technologies to solve the problem of communication failures. Taner et al. (2015) applied the model-mediation method to a mobile robot in a simulation environment. Using a method that focuses on the environmental interactions on a mobile robot that is intended to avoid collisions may seem counter-intuitive. However, a new approach to haptic rendering for mobile robots was realized. Instead of generating force based on the master and the proxy, the distance from the slave to the environment and the slave velocity were considered. This virtual force allows the user to sense the environment without collisions on unlimited-workspace operations.
- Panzirsch et al. (2018) applied the model-mediated approach to a mobile robot in a real-world setup with a twist on the force generation. Fictitious forces were rendered according to the slope of the robot rather than to the distance of obstacles. An haptic joystick was used to assist the user to keep the balance of the vehicle.
- A step towards real-world application of MMT was taken in Liu et al. (2018). A bilateral teleoperation system was developed and tested in a simulation environment to be used in robot-assisted surgery. In this study, a non-linear Hunt-Crossley model approximator (Hunt and Crossley, 1975) was implemented to construct soft tissue contact models in virtual environment. Model parameters are estimated online and the stability of the system is guaranteed through passivity analysis. Simulation results shows that the proposed method is able to match the estimated and the real

environment properties, presenting promising results to be used in real-world applications.

- Valenzuela-Urrutia et al. (2019) proposes a virtual reality-based teleoperation method utilizing point cloud data. In this system, visual representations of the master and the slave, respectively called *robot-shadow* and *robot-real*, are modeled in the virtual environment. This allows the user to see both the virtual and the real robot in the same environment. The operator can interact with the virtual environment and receive haptic feedback without any delay. When the final position of the robot-shadow is decided, the real robot moves to the position of the robot-shadow corresponding in the real environment.

Haptic feedback is generated from the point cloud data obtained from the real environment using an RGB-D camera utilizing the algorithms presented in Ryden et al. (2011) and Ryden and Chizeck (2013). The proposed method was tested in multiple scenarios varying in difficulty and in different feedback types, visuo-haptic and only visual. The results show that the task completion time was lower in the haptic method with more accurate movements and less collisions.

In this chapter, advancements in the field of model-mediated teleoperation were overviewed. The literature survey assisted the design and implementation of the methods used in this study. The following chapter explains this process in detail.

CHAPTER 3

METHODOLOGY

This chapter describes the methods and components used in this study. Methods consist of teleoperation architecture, controllers, and image processing for augmented vision. The components include all hardware utilized.

In the first section, proposed teleoperation method is introduced including the teleoperation structure and haptic rendering on master side.

In section 3.2, hardware of the test setup is explained. The capabilities of the master device and the components of the slave robot is included. The workspace of the robot is defined with the assumptions made to simplify the computations.

Section 3.3 describes the implementation of control system. Information flow between the processes are shown in addition to communication between the master and the slave.

Augmented reality methods are specified in Section 3.4. Image processing algorithm to render augmented reality from real life video feedback are presented.

The experimental procedure and the test results of these sections are covered in the next chapter.

3.1. Proposed Model Mediated Teleoperation Method

The proposed model mediated teleoperation architecture is represented in Figure 3.1. The master and the slave environments are separated by a delayed communication line shown as time delay in the figure. In the master side, the master device, Novint Falcon receives motion commands from the human operator. It is controlled by the master controller. The master controller is responsible for controlling the force output of the Falcon and calculating the position of the end effector. Jacobian transpose is applied to convert the reference force values calculated by the master force interpreter into the reference torque values for the motors. At the same time, forward kinematics is applied to the encoder readings to obtain the end effector position.

In the master force interpreter, the current proxy position and the model of the slave environment is considered to calculate the force values to be directed to the operator.

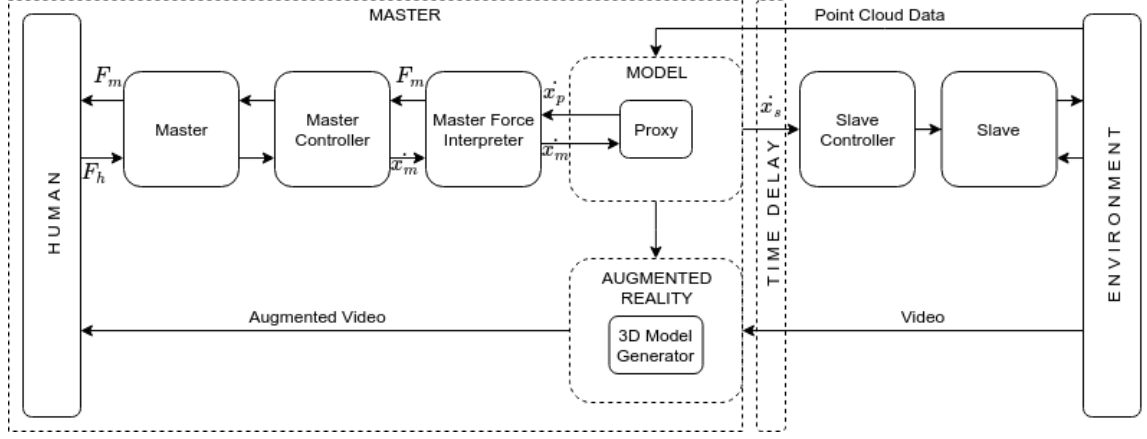


Figure 3.1. Proposed teleoperation scheme.

The original model mediated teleoperation method suggests that the force to be reflected is both proportional to the position and velocity difference between the proxy and the master, shown in Equation 3.1.

$$\bar{F}_m = \hat{K}_{pm}(\bar{x}_p - \bar{x}_m) + \hat{K}_{dm}(\bar{v}_p - \bar{v}_m) \quad (3.1)$$

where \bar{F}_m is the force reflected from the master to the user, \hat{K}_{pm} and \hat{K}_{dm} are PD control gain matrices, \bar{x}_p and \bar{x}_m are respective positions of the proxy and the master, \bar{v}_p and \bar{v}_m are the velocities in the same manner.

The difference is caused by the defined proxy behaviour where the proxy stays just above the surface, as close as it can get without penetrating. However, this method requires the master to touch the virtual environment. Since the proposed method is experimented on a mobile robot as the slave, it is expected for the slave not to interact physically with the environment. The same principle applies for the proxy and the virtual environment as well. Instead, the force is generated in regard to how close the proxy is to the nearby objects.

A critical zone is defined as a circle, and its center is coincident with the slave robot's center. The radius of the zone determines the area in which the haptic rendering occurs. As illustrated in Figure 3.2, state (a) is where no force is reflected to the user since there is no obstacle inside the critical zone. However, in state (b), the obstacle is inside the critical zone. Therefore, the user feels a force that is calculated with respect to the distance between the center of the zone and the closest point of the obstacle. A damping effect is also added to the system as seen in Equation 3.2.

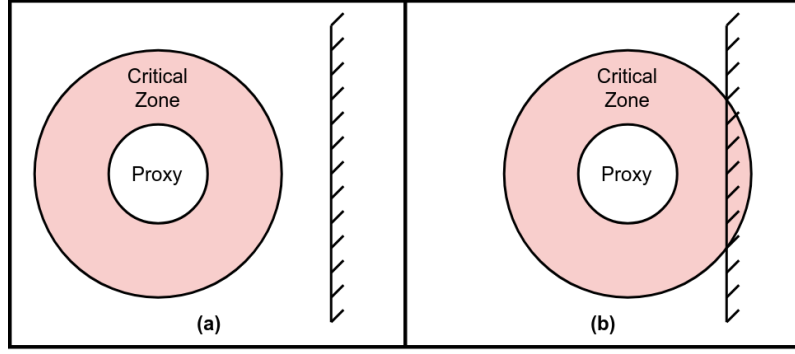


Figure 3.2. Critical zone for haptic rendering.

$$\bar{F}_m = \hat{K}_{pm}(\bar{x}_p - \bar{x}_{cp}) + \hat{K}_{dm}(\bar{v}_p - \bar{v}_{cp}) \quad (3.2)$$

This equation differs from Equation 3.1 by having the position (\bar{x}_{cp}) and velocity (\bar{v}_{cp}) of the closest point instead of the master's. v_{cp} can also be removed from the equation as the obstacles are assumed to be stationary; thus, the velocity is zero. Then, the final equation becomes:

$$\bar{F}_m = \hat{K}_{pm}(\bar{x}_p - \bar{x}_{cp}) + \hat{K}_{dm}\bar{v}_p \quad (3.3)$$

Model subsection of the teleoperation scheme (Fig. 3.1) consists of building the virtual environment from the data coming from the LIDAR sensor on the slave. Hector SLAM ROS package is utilized for building the map and estimating the current position of the robot with respect to the map. Hector SLAM is an open-source set of tools aimed at mapping and localization in closed environments. It is developed by Team Hector (Heterogeneous Cooperating Team Of Robots) of Technical University of Darmstadt. The map is constructed by converting the scan endpoints provided by the LIDAR to point cloud and using scan matching algorithms. This approach is proven to be sufficiently accurate for mobile robots and explained in detail by Kohlbrecher et al. (2011). The resulting map is in the form of an occupancy grid map. It is basically a matrix in which each element corresponds to a small square in the environment. As shown in Figure 3.3, white points are empty, black points are occupied by obstacles, and the gray points are unknown. The white lines penetrating the surface are caused by the sensor's height being higher than the obstacles' and computational errors. The resolution of the occupancy grid map is limited by the resolution of the sensor and chosen to be as small as possible in this application.

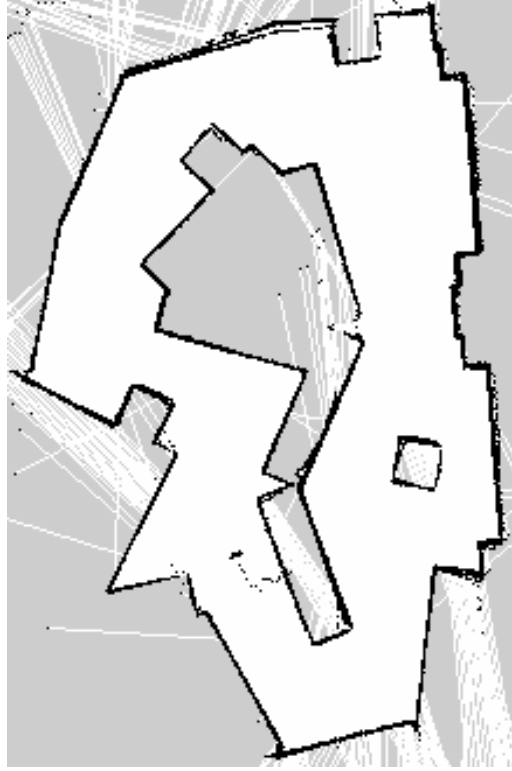


Figure 3.3. An example generated map.

The proxy motion is calculated from the position of the end effector of the master device. Relative position of the master device with respect to its origin is fed to the proxy as velocity command.

$$\bar{v}_p = \hat{K}_v(\bar{x}_m - \bar{x}_0) \quad (3.4)$$

where \bar{v}_p is the proxy velocity, \hat{K}_v is the scaling factor matrix, \bar{x}_m is the master position, and \bar{x}_0 is the origin of the master. Since the proxy is assumed to have no dynamics, the position can easily be calculated by integrating the velocity command. \bar{x}_p is the proxy position and \bar{v}_p is given by Equation 3.4.

$$\bar{x}_p = \int_{t_0}^t \bar{v}_p dt \quad (3.5)$$

The equation is then transformed into discrete form where Δt is the step size, superscripts $t - 1$ and t are the variables in the previous and the current step, respectively.

$$\bar{x}_p^t = \bar{x}_p^{t-1} + \bar{v}_p^t \Delta t \quad (3.6)$$

3.2. Experimental Setup

A teleoperation system is required to validate the proposed method. An experimental setup is built in Iztech Robotics Laboratory (IRL). The test setup is composed of a master and a slave environment. The master environment consists of a PC, and a master device, Novint Falcon. The slave environment includes a workspace defined in Section 3.2.3, and a custom-built omnidirectional mobile robot. The hardware of the master and the slave devices are described in Section 3.2.1, and 3.2.2, respectively. The control methods, teleoperation architecture, image processing, and communication between devices are implemented in Robot Operating System (ROS). The details of the software is discussed in Section 3.3.

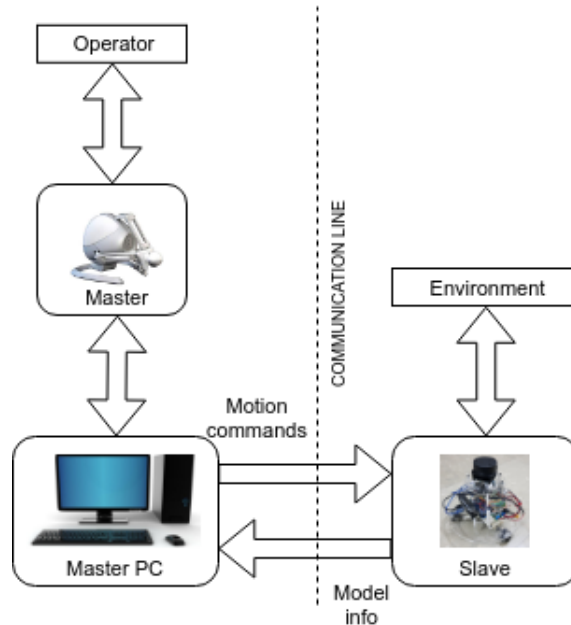


Figure 3.4. Overview of the test setup.

3.2.1. Master Device

Master device is used to send movement commands from the operator to the slave, and force feedback generated from the generated virtual model to the operator. Devices available in the laboratory for this operation are Geomagic Touch X which is a 6-DoF serial haptic device and Novint Falcon, a 3-DoF parallel haptic device. The workspace of

the slave robot is planar. A parallel device with no orientation sensing is more rigid and more suitable for the perception enhancing purposes. Therefore, Novint Falcon is chosen to be used as the master device in this teleoperation.

Novint Falcon (Figure 3.5) is manufactured by Novint Technologies Inc. (NVNT). It is a 3-DoF (all translational) haptic device, and was originally designed as an accessory for PC gaming. Encoders attached to the links of the Falcon determine the position of the end effector with respect to the body of the device. Position of the end effector is used to calculate and send velocity commands to the slave.



Figure 3.5. Novint Falcon desktop haptic device.

Novint is connected to the PC through a USB interface. A cross platform alternative of Novint’s Windows SDK, libnifalcon, is utilized to handle the communications between the device and the ROS platform. Specifications of Novint Falcon are given in Table 3.1

Table 3.1. Technical specifications of Novint Falcon haptic device.

Workspace	10 cm × 10 cm × 10 cm
Force capabilities	9 N
Position resolution	160 increments per cm
Communication interface	USB 2.0
Size	22cm × 28cm × 24cm
Weight	2.8 kg
Power	30 W
Voltage input	100V-240V, 50Hz-60Hz

3.2.2. Slave Device

The slave robot is chosen to be a 3-DoF omnidirectional mobile robot since it is controlled by a three translational DoF haptic device. Planar kinematic constraints are removed so that it is both easier to simulate and to control the position of the robot. This also gives the user a better control over the workspace.

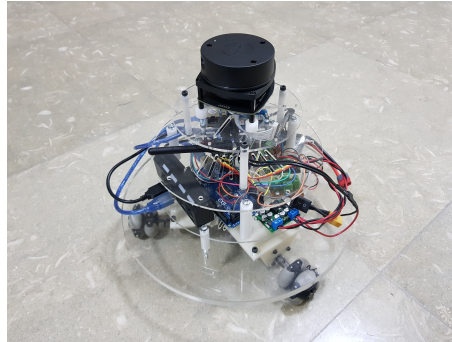


Figure 3.6. The mobile robot.

3.2.2.1. Hardware

The robot (Figure 3.6) is designed and built using the hardware listed below.

- The chasis of the robot is made of plexiglass and has four layers. The motors are attached to the bottom side of the base layer. The electronics are placed on the second layer, the main processor is on the third and the lidar sensor is on the top layer.
- Odroid XU-4 is the main processor unit. It is an 8-core single board computer running on Ubuntu 16.04. It handles wireless communication between the master PC and the slave robot using a wireless adapter. The board is connected to all components such as motor drivers and sensors.
- Arduino Mega2560 is connected to the main processor unit via USB interface. Thus, it is also a part of the ROS network. It acquires the encoder data and sends the data to the Odroid while also receiving motor speed signals to be redirected to motor drivers utilizing pwm signals.

- Three 12V brushed DC motors with a 102.083:1 gear ratio are operated as the actuators of the robot. They are speed controlled by two Pololu dual MC33926 motor driver carriers since each carrier can drive up to two motors. The drivers can deliver up to 3A current to each motor continuously with a peak value of 5A.
- Each motor has an integrated quadrature encoder. The encoders provide a resolution of 64 counts per revolution of the motor shaft. Therefore, the output resolution of the gearbox shaft corresponds to 6533 counts per revolution.
- The output shafts of the motors are connected to three universal omnidirectional wheels. These wheels generate traction in the direction of rotation but allows movement in perpendicular direction.
- The robot is powered by a 11.1V 3S 3800 mAh LiPo battery. A 5V/15A step-down voltage regulator is used to power the Odroid and the Arduino.
- A LIDAR(Laser Imaging Detection and Ranging) device is mounted on the top layer of the robot to scan the environment as well as to estimate the current position of the robot. RPLidar A1M8 360°laser range scanner is a cost-effective alternative among available options. It has a range of 12 m with 0.2 cm resolution. The sensor can provide up to 8000 samples per seconds at 10 Hz full scan rate.

3.2.2.2. Kinematics

In this section, the necessary equations to control the velocity of the mobile robot are derived. Forward and inverse kinematics of the robot are defined to be used by the controllers.

The geometrical representation of the mobile robot is shown in Figure 3.7. The axes of the task space coordinate frame is shown by x_w and y_w . The body-fixed coordinate frame of the slave robot is placed at the geometrical center and represented by x_s and y_s . y_s -axis is defined to be the forward direction and coincident with the axis of rotation of the first wheel. The remaining wheels are evenly spaced around z -axis. Linear velocities of the vehicle are V_x and V_y along the respective axes of the body-fixed frame. The angular velocity around z -axis is ω . Linear velocities of the wheels along the respective axis of traction are V_1 , V_2 , and V_3 . The forward kinematics equations are given in Equation 3.7, 3.8, and 3.9.

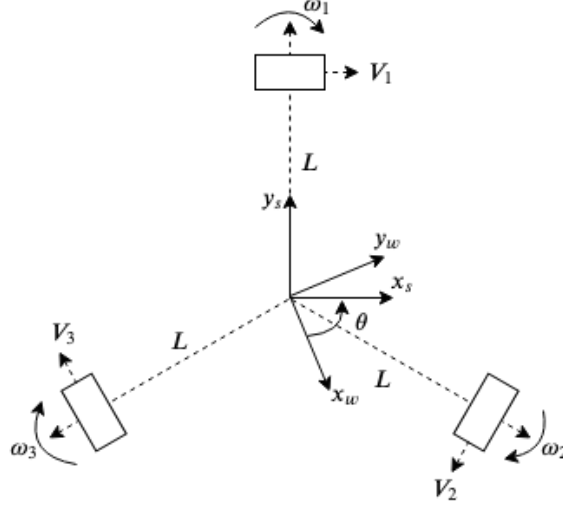


Figure 3.7. Geometrical representation of the robot.

$$V_x = V_1 + V_2 \cos\left(\frac{2\pi}{3}\right) + V_3 \cos\left(\frac{4\pi}{3}\right) \quad (3.7)$$

$$V_y = V_2 \sin\left(\frac{2\pi}{3}\right) + V_3 \sin\left(\frac{4\pi}{3}\right) \quad (3.8)$$

$$\omega = (V_1 + V_2 + V_3)/L \quad (3.9)$$

The matrix representation of these equations are given in Equation 3.10.

$$\begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix} = \begin{bmatrix} 1 & \cos\left(\frac{2\pi}{3}\right) & \cos\left(\frac{4\pi}{3}\right) \\ 0 & \sin\left(\frac{2\pi}{3}\right) & \sin\left(\frac{4\pi}{3}\right) \\ 1/L & 1/L & 1/L \end{bmatrix} \cdot \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} \quad (3.10)$$

Jacobian matrix, \hat{J} , relating the joint space velocities to the task space velocities is a constant 3 by 3 matrix since the task space velocities are defined in body fixed frame of the slave (Equation 3.11).

$$\hat{J} = \begin{bmatrix} 1 & \cos\left(\frac{2\pi}{3}\right) & \cos\left(\frac{4\pi}{3}\right) \\ 0 & \sin\left(\frac{2\pi}{3}\right) & \sin\left(\frac{4\pi}{3}\right) \\ 1/L & 1/L & 1/L \end{bmatrix} \quad (3.11)$$

The inverse of the \hat{J} matrix converts linear robot velocities to linear wheel velocities (Equation 3.12).

$$\begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} = \hat{J}^{-1} \begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix} \quad (3.12)$$

In order to transform required task space velocities to joint space velocities, inverse of the Jacobian matrix presented by Equation 3.11 is used.

$$\hat{J}^{-1} = \begin{bmatrix} \frac{2}{3} & 0 & \frac{L}{3} \\ -\frac{1}{3} & \frac{\sqrt{3}}{2} & \frac{L}{3} \\ -\frac{1}{3} & -\frac{\sqrt{3}}{2} & \frac{L}{3} \end{bmatrix} \quad (3.13)$$

The transformation between the angular and the linear velocities of the wheels are given by the following equations (Equation 3.14).

$$V_i = \omega_i r, \quad \omega_i = V_i / r \quad (3.14)$$

The transformation from world to slave's body-fixed frame is a single rotation around z -axis. Therefore, the angular velocity of the robot is equal in both frames. As seen from Equation 3.10, the velocity of the robot is composed two linear and one angular velocity. The resulting matrix is constructed in Equation 3.15.

$$\begin{bmatrix} V_x^{(s)} \\ V_y^{(s)} \\ \omega \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_x^{(w)} \\ V_y^{(w)} \\ \omega \end{bmatrix} \quad (3.15)$$

3.2.3. Workspace Requirements

The workspace of the robot must be defined clearly since the environment is also modeled virtually. Some definitions and assumptions are made to simplify the implementation of the model mediated teleoperation method.

- The workspace is chosen to be an empty room with several obstacles. Everything in the slave environment except the slave robot is rigid and static. This reduces the requirement for continuous model updates. After the model is built, it is sent to the master once, reducing the online computations and required wireless communication bandwidth.

- The heights of all objects are greater than 25 cm. As the height of the sensor from the ground is also 25 cm, it is guaranteed that the sensor is able to detect all the possible collisions.

3.3. Control System Implementation

Robot Operating System is utilized throughout this study. There are a lot of sub-tasks required to be done during the teleoperation such as acquiring data from the sensors, processing the sensor data, and controlling the motors. Each subtask is run as a separate executable and referred as nodes in the ROS network. The function of the nodes, the communication among them, and the top level teleoperation architecture is explained in this section.

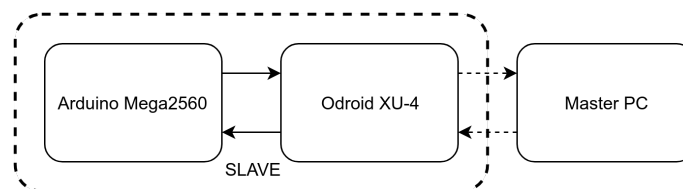


Figure 3.8. ROS node locations

The nodes are run on different hardware depending on the physical location of the task and the involved components as seen in Fig. 3.8. Arduino Mega2560 is in control of a single node that acts as a data acquisition board. It is connected to the motor drivers, the encoders, and the Odroid XU-4. The outputs of three quadrature encoders are handled and sent to the main processor unit as the feedback in the control loop. In response, control output of the motor controllers are received from the Odroid and transmitted to the drivers as pwm signals.

Control algorithms are run on the main processor, Odroid XU-4. Figure 3.9 represents information flow between the nodes. There are two control loops responsible for the position control of the robot. A high level position controller generates reference velocity to a low level motor controller. Position controller receives reference input from the master PC and the feedback from the pose estimation node. Inverse kinematics is applied to the output of the controller, and sent to the motor controller as reference input. Motor controller node gets the feedback from the Arduino node and returns the control output back.

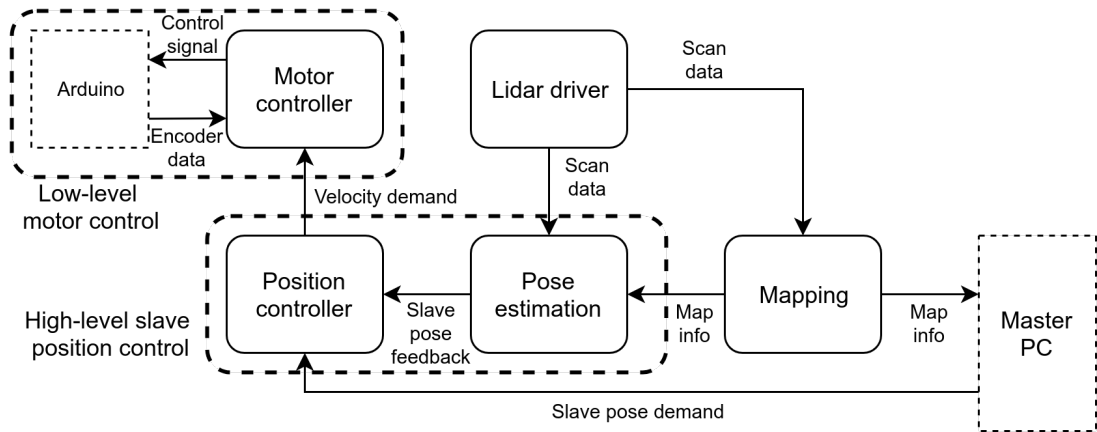


Figure 3.9. Illustration of ROS nodes running on Odroid.

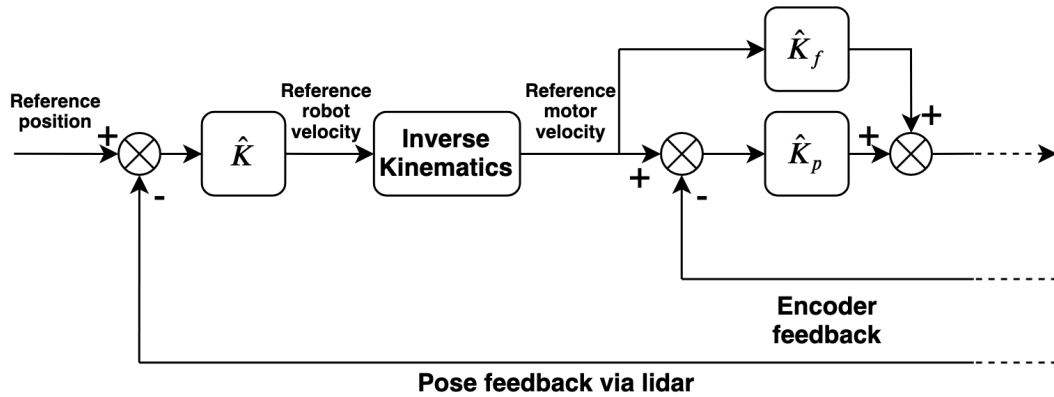


Figure 3.10. High level position and low level velocity controllers.

Figure 3.10 shows the block diagram of the controllers. The relation between the desired velocity and the input voltage supplied to the motors is measured via experimental tests. This relationship is found to be a linear one within the input voltage range of 2.4 - 12 VDC. This linear relation (presented in the Appendix) is mapped to a feedforward velocity gain (\hat{K}_f) in the low-level controller for wheel velocities. Feedforward term allows faster reactions by estimating required input to the motors (Franklin et al., 1998). A proportional control with a constant gain matrix (\hat{K}_p) is used. This gain value is experimentally tuned to compensate for the inaccuracies caused by an assumed linearity while determining the feedforward gain.

High-level controller is selected to be a proportional controller since the required velocity to compensate for a given position error is proportional to magnitude of the position error. Initially, gain matrix (\hat{K}) is estimated with respect to the sample time of the position estimator and the maximum velocity of the slave robot as shown in Figure

3.16. Then, it is tuned experimentally until the position tracking performance is within the benchmark values specified in Rowekamper et al. (2012).

$$V_{required} = \frac{e}{\Delta t} \quad (3.16)$$

where $V_{required}$ is the velocity value needed to compensate for the position error, e in one time step, Δt .

The master PC is responsible for human computer interaction (Figure 3.11). Since the slave is a mobile robot, position-to-position mapping between the master and the slave is not feasible. Therefore, position of the master device should be mapped to the velocity of the proxy as described in Farkhatdinov and Ryu (2007).

In this study, Novint Falcon is used as a planar two-axis master system by constraining its z-axis motion via control. Novint Falcon driver supplies its measured task-space position on a horizontal plane to the proxy pose generator. In this way, both task spaces of the master, proxy and the slave have two DoF. This position information is amplified by a gain to increase the operational speed of the proxy and it is received as a velocity demand by the proxy. A dead-zone is defined at the origin position of the Novint Falcon's task space to prevent undesired movements when the user tries to send zero velocity demand. The proxy pose is calculated by integrating this velocity information. This pose is then forwarded to the Odroid with an artificially induced delay to be used in high level controller represented in Figure 3.10.

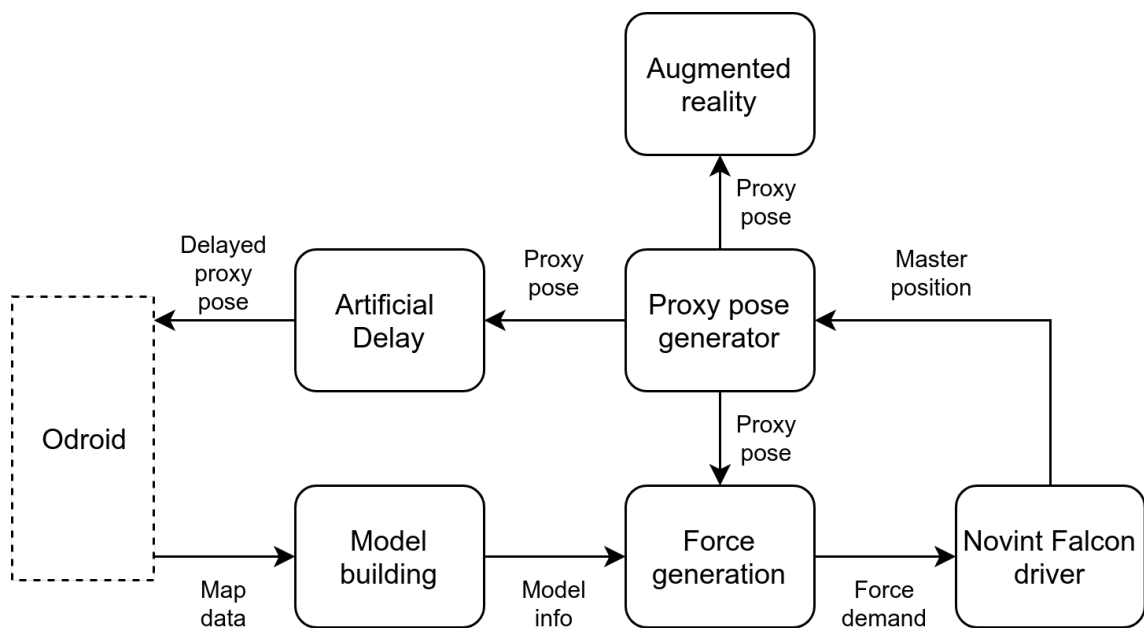


Figure 3.11. Illustration of ROS nodes running on the master PC.

Once the proxy pose is calculated, this information is used augmented reality node to render the visual feedback to the user. In parallel, the force generation node calculates the force to be displayed to the user. This is accomplished by comparing the proxy pose and the model information received from the slave. Then, this calculated force information is sent to the Novint Falcon driver. The calculations for force generation are explained in section 3.1.

3.4. Augmented Reality

The proposed extension to the model mediated teleoperation method involves an augmented reality based visual feedback to the user. In this section, implementation of augmented reality is described in detail. OpenCV, an open source image processing library, is utilized to modify the images captured by Logitech C103 camera. Virtual model of the slave robot is rendered with respect to the camera position and orientation, to be merged with the camera image. The 3D model of the robot is simplified to reduce the required computational power and to keep the frame rate above 30 (Figure 3.12).

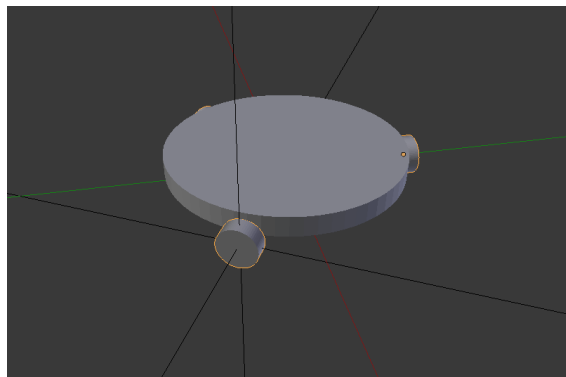


Figure 3.12. The 3D model of the slave robot.

Rendering process is done by projecting 3D points onto the image plane. The 3D points referred in this context are the points forming the 3D model of the slave robot. The pinhole camera model (Figure 3.13) is used to transform the points into 2D coordinate frame, referred as the image plane. In pinhole camera model, geometrical projection of a pinhole camera is mimicked but the entire optics and the hole is reduced to a single point called the optical center or the center of projection. The following terms are introduced to describe the algebraic formulation of the model. The origin of the camera coordinate system is coincident with the optical center. The line passing through the optical center

and is also orthogonal to the image plane is the optical axis. The optical axis is coincident with the Z-axis of the camera coordinates. The intersection of the optical axis and the image plane is the principal point. The principal point is also the origin of the image coordinate system. The x/y-axes of the image coordinate system and the camera coordinate system are parallel. The distance between the center of projection and the image plane is the focal length.

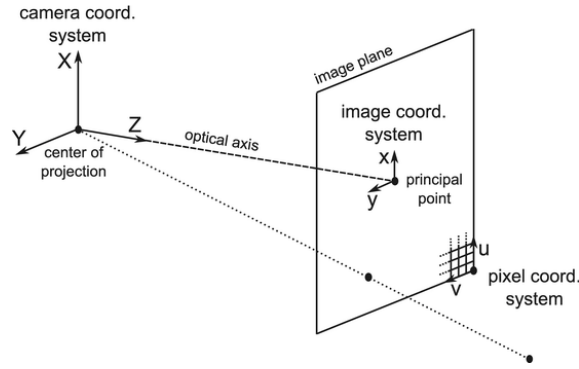


Figure 3.13. The pinhole camera model.

Even though these geometrical definitions are sufficient to explain the pinhole camera model, a 2D coordinate system must be defined to display the final digital image. This coordinate system is called pixel coordinate system and the axes are defined as u and v . It represents the layout of the pixels of the digital image. The u and v -axes are also parallel to the x/y -axes of the previously mentioned coordinate systems. The origin is placed at the lower-left corner of the image plane while viewed from the optical center and denoted by $(-x_0, -y_0)$.

In order to find where a 3D point corresponds to in the image, a straight line is drawn from the point to the center of projection. The intersection of this line and the image plane gives the coordinates of the point in the image coordinate system. Defining the coordinates of a 3D point as (X, Y, Z) , the projection of this point onto the image plane is calculated from the comparison of similar triangles in Equation 3.17 (Sturm, 2014).

$$x = f \frac{X}{Z}, \quad y = f \frac{Y}{Z} \quad (3.17)$$

The point is shifted by (x_0, y_0) to match the origins. A scaling of k_u and k_v is applied to convert from metric units to pixel units where f is the focal length, k_u and k_v are the density of pixels in u and v directions, respectively.

$$u = k_u(x + x_0), \quad v = k_v(y + y_0) \quad (3.18)$$

Substituting equation 3.17 into equation 3.18:

$$x = k_u f \frac{X}{Z} + k_u x_o, \quad y = k_v f \frac{Y}{Z} + k_v y_o \quad (3.19)$$

The equation is expressed in homogeneous matrix representation to be used in computations:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} k_u f & 0 & k_u x_o & 0 \\ 0 & k_v f & k_v y_o & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

k_u , k_v , f , x_o , and y_o are replaced by the pixel unit equivalents. The upper triangle of the resulting matrix, K , is called the *calibration matrix* and the non-binary elements are the *intrinsic parameters* of the pinhole camera model.

$$\hat{K} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.20)$$

The knowledge of the pose of the camera is required if the camera is moving. In order to describe the pose, a *world coordinate system* is defined with respect to an object with a known constant pose. A 3D point is transformed from the world to the camera coordinate system with the following matrix equation:

$$\begin{bmatrix} X^{(c)} \\ Y^{(c)} \\ Z^{(c)} \\ 1 \end{bmatrix} = \begin{bmatrix} \hat{C} & -\hat{C}\bar{T} \\ \bar{0}^T & 1 \end{bmatrix} \begin{bmatrix} X^{(w)} \\ Y^{(w)} \\ Z^{(w)} \\ 1 \end{bmatrix} \quad (3.21)$$

where \hat{C} is the rotation matrix defining the orientation of the camera, \bar{T} is the translation vector defining the position of the camera, and the superscripts c and w represents the coordinates of the point in the camera and the world frame, respectively.

There are several changes made to implement the pinhole camera system to the proposed augmented reality based teleoperation. In the teleoperation, the camera is stationary and the 3D model moves with respect to the camera frame. The 3D model of the robot is defined in its own coordinate system. Since converting every point in the model to the world coordinate system would be computationally expensive, the camera pose is defined in the robot coordinate system. In this case, even though the robot is moving, it is assumed to be idle at the origin of the world coordinate system. Instead, the camera

moves in the opposite direction of the robot. The perspective from the camera to the robot is exactly the same as it would be in the original case.

CHAPTER 4

EXPERIMENTAL TESTS

Performance of the controllers and the validity of the proposed method is tested through a series of experiments. Initial set of experiments investigates the accuracy of the augmented reality and the performance of both high- and low-level controllers.

After the integration of the system is validated, the tests with human subjects to evaluate the performance change of the teleoperation and the telepresence are conducted.

The equipment and methods are explained and received measurements are presented and discussed in this section.

4.1. Augmented Reality Tests

Augmented reality tests were done to both calibrate the camera and match the scaling between the virtual objects and the real world. A scaling and an offset is applied to neglect the effects of camera distortion and insufficient measurement accuracy of the camera's slope, shown in Equation 4.1.



Figure 4.1. Uncalibrated augmented vision.

$$x'_p = \hat{K}_s x_p + \bar{s}_o \quad (4.1)$$

where x_p is proxy position, x'_p is calibrated proxy position \hat{K}_s is a diagonal scaling matrix for x and y axes, and \bar{s}_o is a column vector containing x and y offset values.

An initial scaling and offset is given as a starting point for iterations. Augmented image is generated according to reference points. An example of an uncalibrated augmented vision is shown in Figure 4.1.

After a set of reference locations are selected, the virtual object is rendered on these locations with the initial parameters. Reference locations are marked with squares as shown in Figure 4.1 and Figure 4.2. The points that the rendered points correspond in real life environment are recorded and the experiment continues with a new set of parameters selected according to the error in placements.



Figure 4.2. Calibrated augmented vision.

Iterations are continued until all of the augmented robots are placed inside the reference squares.

4.2. Teleoperation Performance Tests

The proposed method is evaluated with the performance evaluation tests. Since the telepresence is directly related to the sense of human users, the tests were performed on human subjects as the operators of the system. The tests involve ten people and three scenarios. Written consent was given by all subjects prior to the tests. In each scenario, time delay of 1 second is introduced in both directions of communication, from master to slave and from slave to master. Additionally, users are provided with a live video feedback

that has a communication delay of 1 second. In order to standardize the evaluation, four checkpoints are marked on the floor in a square formation as seen in Figure 4.3.

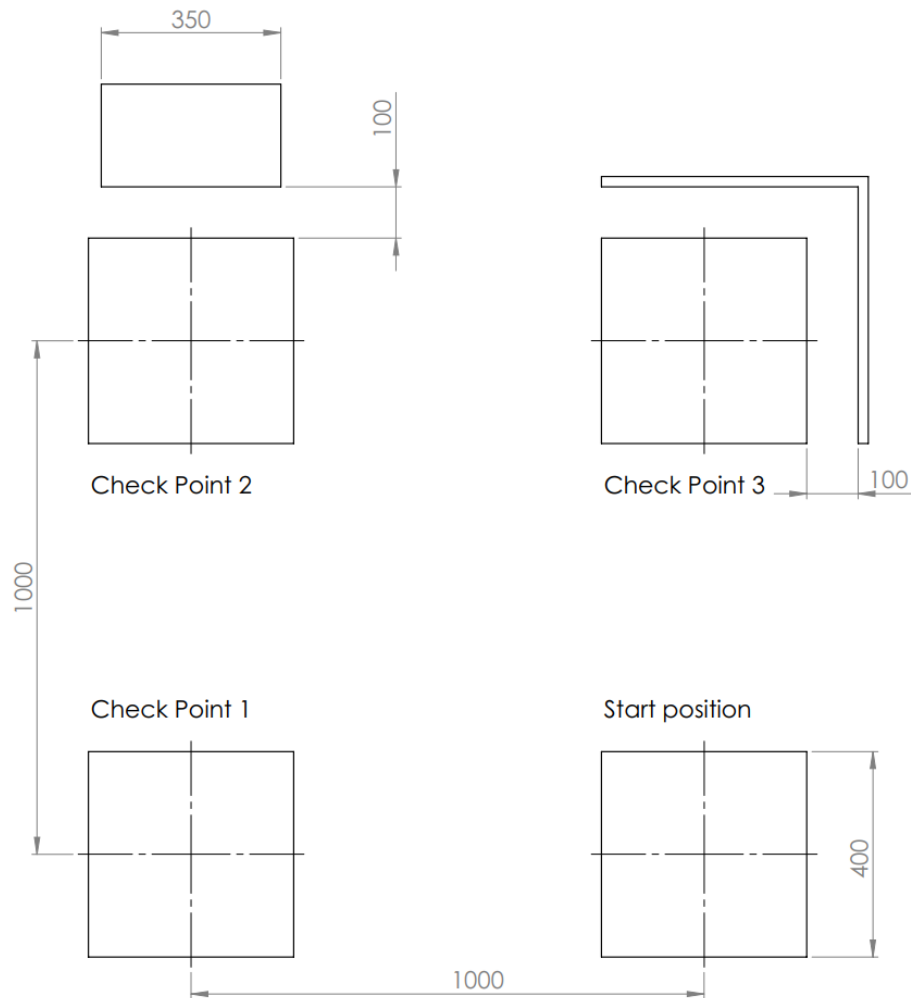


Figure 4.3. Bilateral teleoperation experiment setup.

Before each test, users are given sufficient time to get used to controlling the robot using all three methods. In the beginning of the procedure, the robot starts at the bottom-right corner in camera point of view. The user is asked to visit each checkpoint in clockwise direction, ending at the starting point.

The first test is standard bilateral teleoperation, shown in Figure 4.4 without any assistance or method to prevent the effects of time delay. Since the stability of the system is not guaranteed in this method, the users are expected to have difficulties completing the tasks.

In second set of tests (Figure 4.5), model-mediated teleoperation approach is applied to the system. Subjects are only provided with the visual feedback from the virtual



Figure 4.4. Bilateral teleoperation experiment setup.

environment, therefore, they cannot see the real robot in action. Subjects are asked to move the virtual robot visiting the checkpoints in a virtual environment that is identical to the real slave environment. A checkpoint is accepted to be completed if the virtual robot stays inside the marked area for 2 seconds without exiting. While inside the checkpoint borders, the color of the robot changes to yellow as an indication to the user. If the user manages to complete the checkpoint, the color changes to green for 2 seconds. Then, the user can continue with the next checkpoint. This procedure is repeated for each point until the last checkpoint located at the starting position is completed. The color changes to blue to let the user know that the test is complete.

For the last test, model-mediated teleoperation is enhanced with augmented reality as shown in Figure 4.6. The checkpoints are exactly the same as the first two tests. Aside from delayed video feedback, the subjects are provided with the representation of the virtual robot. The color of the virtual behaves as previously defined in the second test. This allows user to have a knowledge about the progress of the test.

After three tests for each subject, a set of questions are asked to evaluate the effects of the proposed method. The questions are as follows:

- Which method is the easiest to use?
- Which method is the safest to use?
- Which method has the highest sense of accomplishment for the tasks?
- Which method has the highest telepresence?

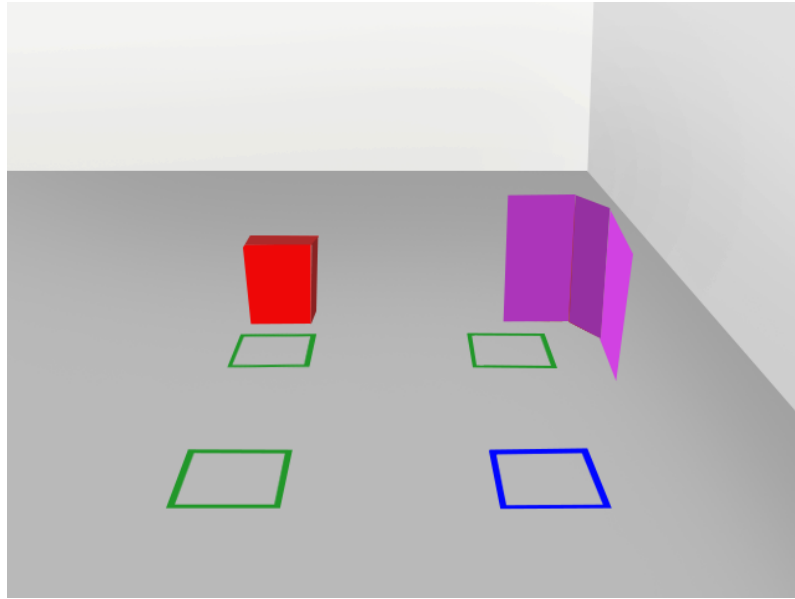


Figure 4.5. Virtual reality based model mediated teleoperation test setup.



Figure 4.6. Augmented reality based model mediated teleoperation test setup.

- Which method do you prefer to control the robot?

The results of the tests are presented and evaluated in the next section.

4.3. Teleoperation Performance Test Results

The performance of the teleoperation is evaluated in terms of position tracking of the slave robot, performance comparison of the proposed method, and the feedback from the subjects.

The results obtained from the tests are presented. The original and the proposed method is compared with respect to previously mentioned categories.

4.3.1. Controller Performance Results

Position tracking capabilities of the mobile robot are investigated in this section. The position of the proxy robot is the reference input for the position controller. During tests, both proxy position and real robot position are recorded and compared.

Figure 4.7 shows x-axis position of proxy and slave robot during teleoperation. Position of the proxy is generated from the master device and displayed to the user without any delay. However, due to 1 s time delay in communication, position input transmitted to the slave is delayed. Position tracking delay can be determined by correlating the positions and shifting the graph accordingly.

Matching of the positions is established when the proxy position is moved 2s forward (Figure 4.8). Since the robot moves to the starting position autonomously before each test, a small starting error occurs. Error is compensated after the teleoperation begins. The difference of 1 s extra delay is caused by the dynamics of the slave robot while the status of the robot changes from stationary to moving. This can also be observed between 20-80 s when the movement direction changes.

Y-axis positions of the proxy and the slave are represented in Figure 4.9. As in x-axis, 2 s time delay in position tracking is observed in y-axis. When the proxy position is shifted forward by 2 s, position matching is achieved (Figure 4.10).

Position tracking error graphs of x- and y-axes are shown in Figure 4.11 and Figure 4.12, respectively.

Mean and standard deviation values of x, y and total errors are presented in Table 4.1. The test results show that the position tracking error of the robot is within the range

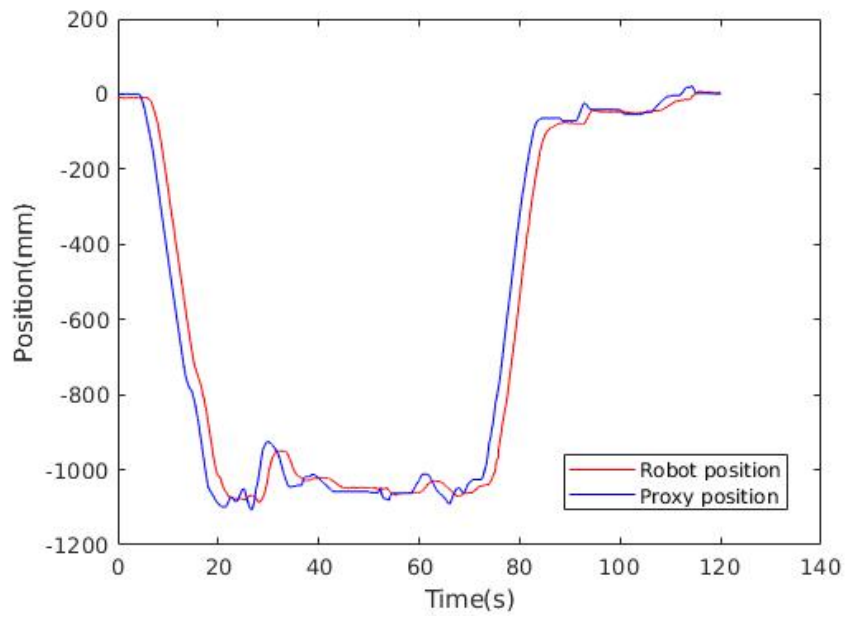


Figure 4.7. Proxy and robot positions on x-axis of the world frame

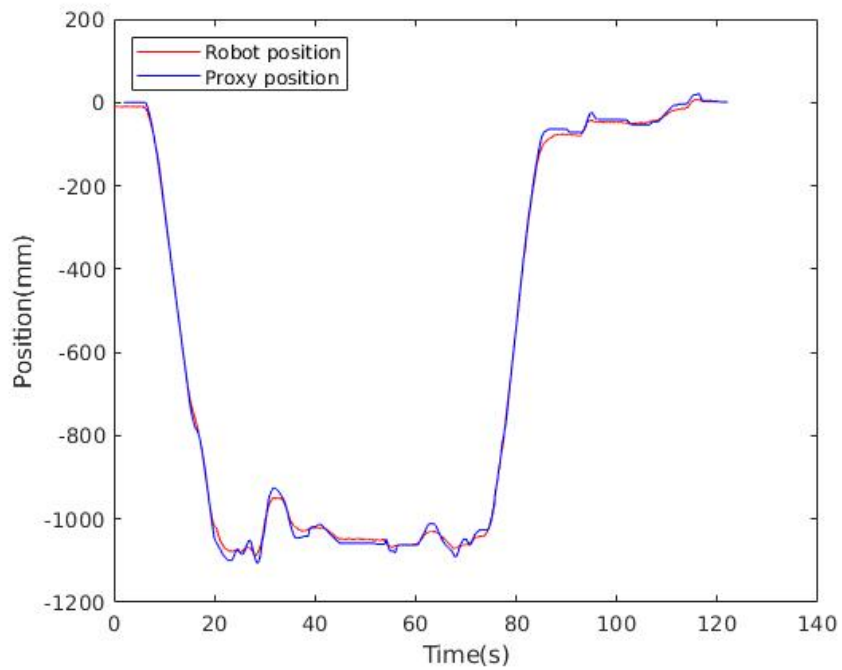


Figure 4.8. Matched proxy and robot positions on x-axis of the world frame

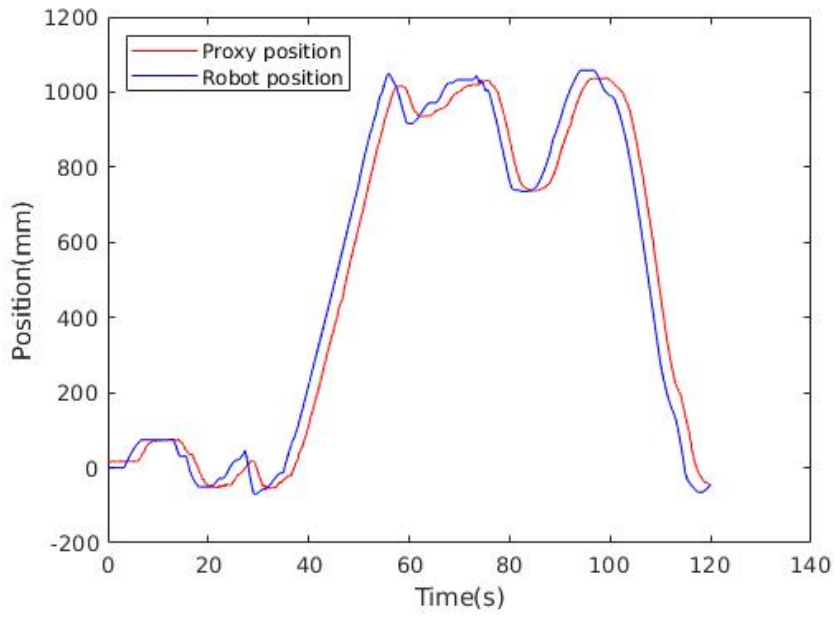


Figure 4.9. Proxy and robot positions on y-axis of the world frame

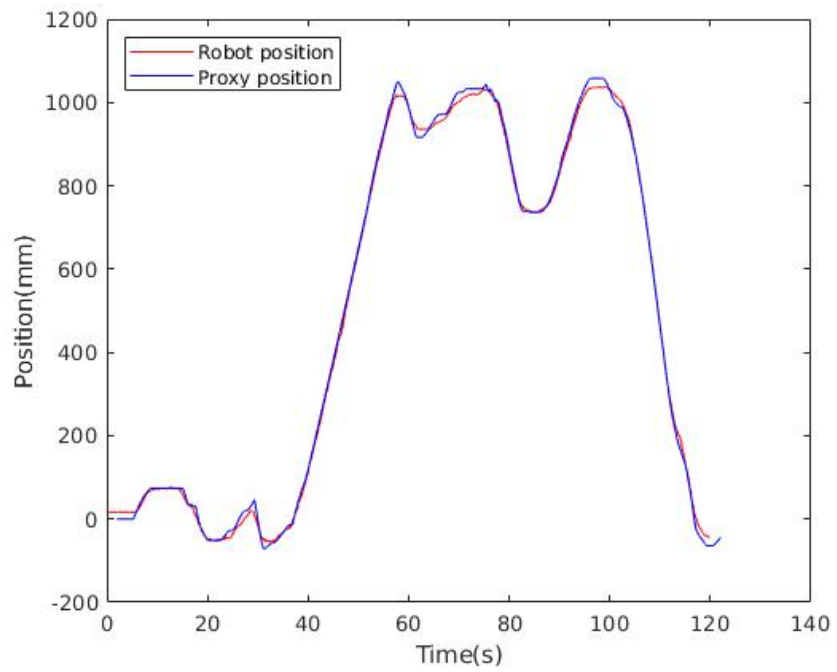


Figure 4.10. Matched proxy and robot positions on y-axis of the world frame

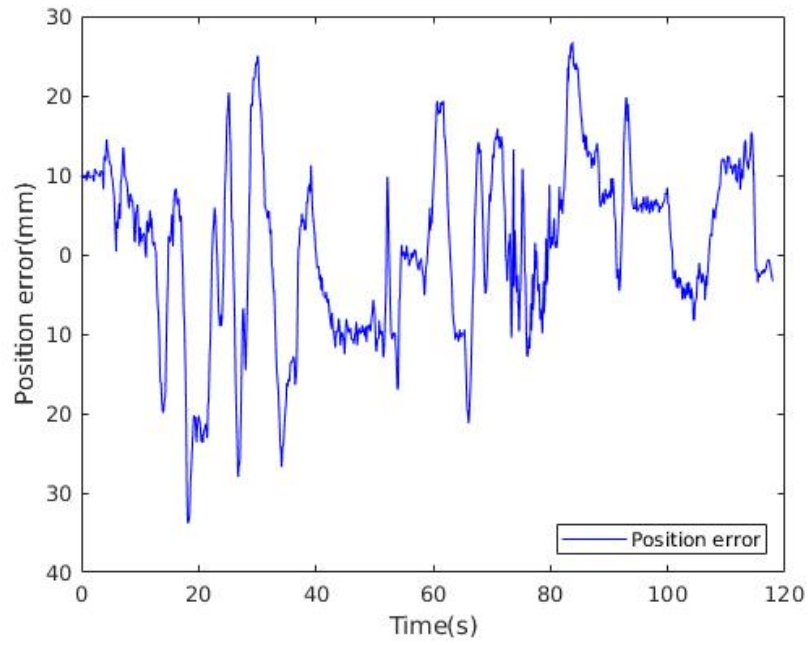


Figure 4.11. Position tracking error in x-axis of the world frame

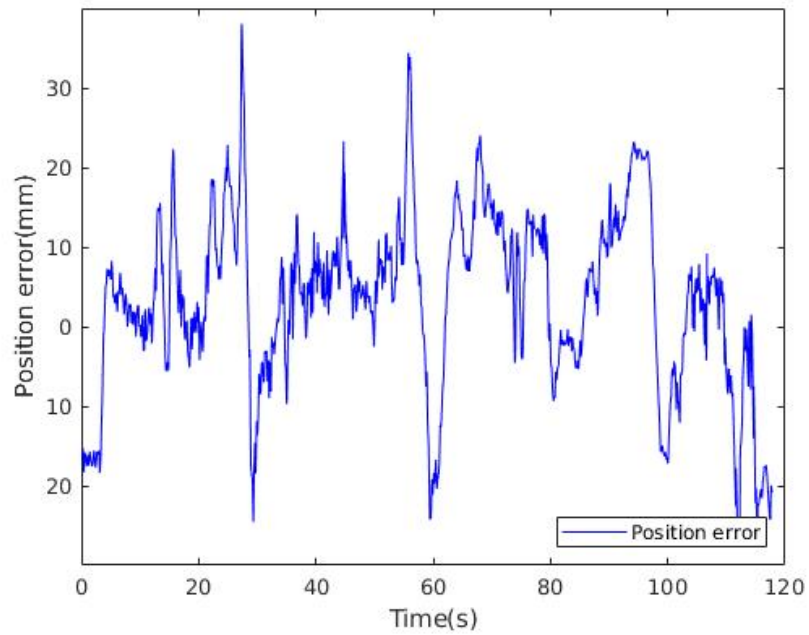


Figure 4.12. Position tracking error in y-axis of the world frame

Table 4.1. Position tracking errors in x/y-axes and in total.

	x-axis	y-axis	total
Mean	8.9mm	9.3mm	14.3mm
Standard deviation	6.4mm	6.9mm	7.0mm

of similar studies on mobile robot applications (Rowekamper et al., 2012).

4.3.2. User Test Results

User tests consist of completing the same predefined tasks in three different teleoperation methods, described in Section 4.2. Results of the user tests are evaluated in terms of test completion time, accuracy inside the checkpoints and feedback questions.

Table 4.2. Completion times for each subject and test

	No Method	Virtual Reality	Augmented Reality
1	164.601294	94.9998039	134.53225
2	125.232936	114.233416	123.86672
3	117.000289	103.966314	114.399523
4	110.999913	80.8007101	104.57239
5	141.73326	104.033519	149.699011
6	120.100985	74.733367	75.3339277
7	124.898562	104.666264	66.866871
8	112.167515	74.5688993	116.536065
9	102.300287	84.5023434	116.767489
10	110.932704	76.5008709	74.7337441
mean	122.996775	91.3005508	107.730799
std dev	18.1660138	14.7975436	27.4102211

Test completion times are shown in Table 4.2. As seen from the table, standard bilateral teleoperation without any compensation for time delay has the highest mean completion time. This result is expected since both the force and the video feedbacks are delayed. The subjects tend to use move-and-wait method to control the robot. The completion time in this method is proportional to communication delay in the system (Ferrell, 1965).

The lowest mean completion time belongs to virtual reality test. This result is also expected because the subjects are provided with the proxy position which is not affected by the communication delay. Delayed force feedback problem is also not present in this

method.

Augmented reality based teleoperation method has the second lowest mean completion time. Compared to virtual reality method, this method has additional sensory feedback from the slave environment. Since the real robot can be seen while performing the tasks with the virtual proxy robot, the subjects tend to wait for the real robot to arrive at the checkpoint after the checkpoint is completed by the virtual robot. This allows the users to periodically check the motion of the real robot without compromising the telepresence.

Aside from completion time, proxy and robot trajectories are also investigated for each test. The data of the user whose results are the closest to the mean results are shown in Figure 4.13, Figure 4.14, and Figure 4.15 for standard teleoperation method, virtual reality method, and augmented reality method, respectively.

In the beginning of each test, the haptic device needs to calibrate its position. Calibration is done by moving the end effector to the outer edge of its workspace and back in. Because of this, the teleoperation begins with a slight velocity in forward direction. Consequently, the curved motion between the starting point (lower-right corner) and the first checkpoint(lower-right corner) is observed for all methods, shown in trajectory graphs.

In Figure 4.13, due to communication delay, users tend to miss and overshoot the checkpoints as it can be observed at the corners of the graph. Delayed video feedback causes the users to keep moving even though the robot has already reached the checkpoint. This situation does not only exist for checkpoints. The errors in position takes some time to correct. Therefore, it prevents continuous motion during teleoperation. Constantly moving and waiting for the robot to reach the desired point both increases the operation time and the position error due to wind-up time of the slave.

In Figure 4.14, trajectories for VR test is presented. In this method, subjects are completely isolated from the real environment and effects of delayed communications is not reflected. Controlling the virtual robot without any delay allows subjects to correct errors more easily. The second checkpoint, upper-left corner, is where the users receive haptic feedback from the environment for the first time. Therefore, a movement in negative y-direction is observed due to force feedback. For the third checkpoint, upper-right corner, a similar behavior is recognized in both negative x- and y-directions since the obstacle covers the whole corner area.

Figure 4.15 shows the trajectories for AR test. The users have visual feedback from both virtual and real robot. As in VR method, the users can control the virtual robot without any delay. They can also observe the behavior of the real robot. The checkpoints 1

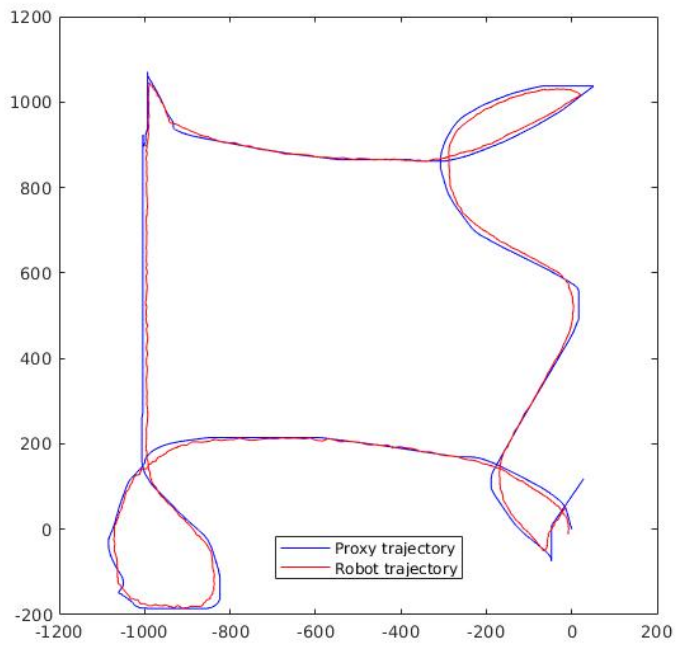


Figure 4.13. Proxy and robot trajectories for teleoperation test.

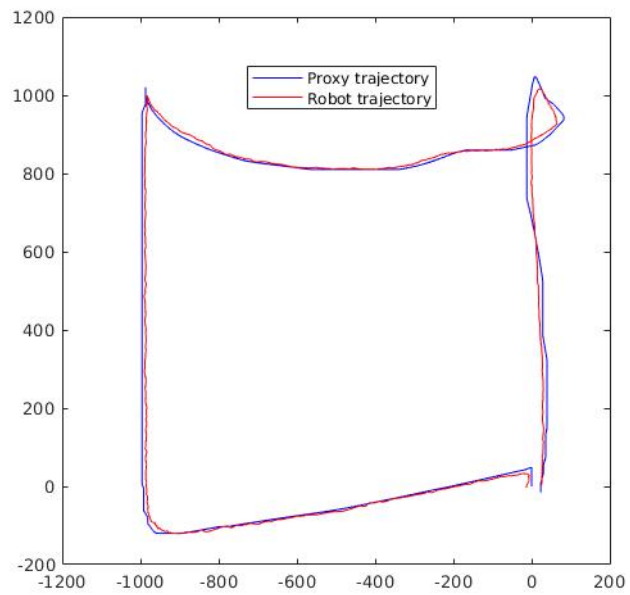


Figure 4.14. Proxy and robot trajectories for VR test.

and 2 shows that the users spend more time inside the checkpoints, waiting for real robot to arrive. This also affects the sense of accomplishment, discussed in detail in Section 4.3.3.

After evaluating the trajectories for all three methods, checkpoint accuracy of the subjects for each test are criticized. Accuracy is the mean value of the robot's position in the time interval, starting from the first time the robot enters a checkpoint area until that checkpoint is completed. Mean point gives an estimation about how difficult it was to complete a checkpoint.

Figure 4.16 shows that the subjects had difficulties reaching and staying inside the checkpoint areas. Compared to Figure 4.17 and 4.18, the points are more spread out. This corresponds to lower precision compared to other two methods. Lower precision among all test subjects implies that the method highly depends on the skill of the user. However, taking Figure 4.17 and 4.18 into consideration, points are more condensed. Therefore, the precision is higher. This indicates that standard and augmented reality enhanced model mediated teleoperation methods are independent of user skill, making these methods reliable and higher performance.

Analyzing the second checkpoint in standard teleoperation (Figure 4.16), the points are closer to the obstacle because of the overshoot caused by communication delay. In Figure 4.17 and 4.18, the points are closely packed near the checkpoints. Real time force feedback increases the telepresence of the users, making it easier to complete the task. However, at the third checkpoint due to the corner obstacle, users experience excessive force in the opposite direction of the obstacle. Reflection of this excessive force at the master side causes the end effector of the haptic device to move in the direction of the force and becomes harder to navigate the robot. This situation also occurs for the second and the third methods. Since the force feedback in these methods are not delayed, mean positions of the robot is closer to the checkpoint. These observations are consistent with the expectations before the tests, and shows that the stability of teleoperation is higher in model mediated teleoperation methods.

Figure 4.19 merges Figure 4.16, Figure 4.17, and Figure 4.18 observe each checkpoint for each method in a more compact form. Bar graphs show the average of mean distances for all subjects. In other words, if the motion of the robot is assumed to be in a circular area near the checkpoint, the mean value gives the distance between the center of this area and the checkpoint. Standard deviation value gives the radius of the circle. This can be used to evaluate the difficulty of each task. Across all checkpoints, it appears that the second one is the easiest due to the obstacle being right behind the checkpoint in the

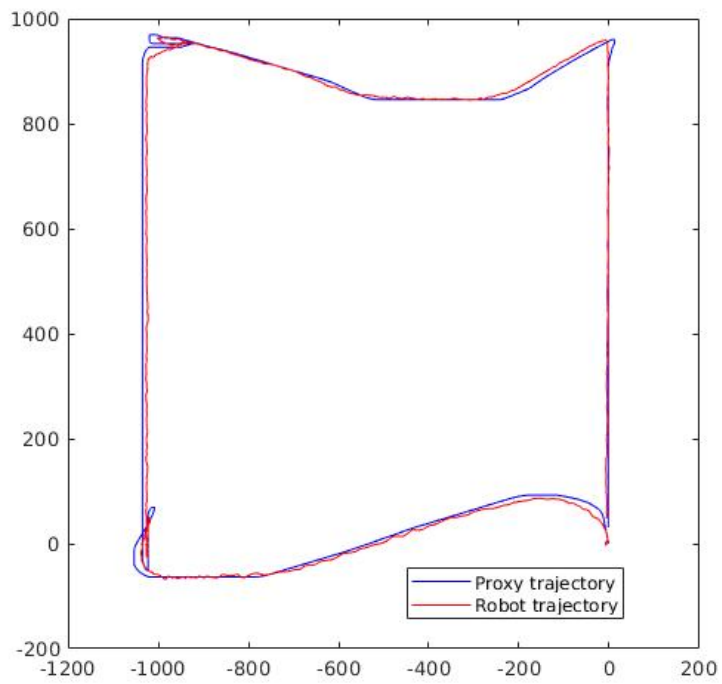


Figure 4.15. Proxy and robot trajectories for AR test.

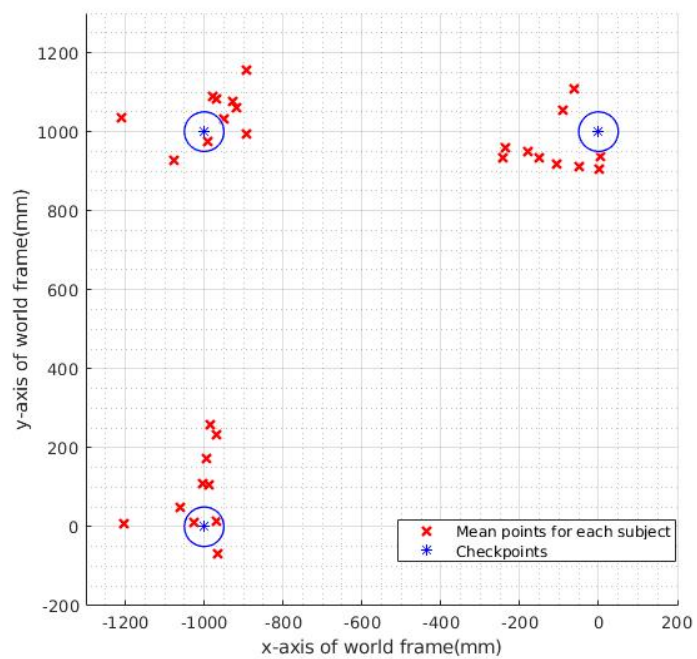


Figure 4.16. Mean position of the robot while completing the checkpoints during standard teleoperation test.

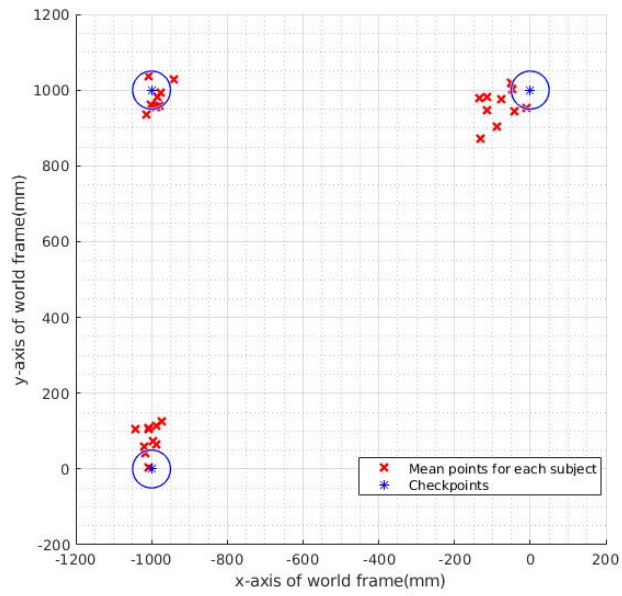


Figure 4.17. Mean position of the robot while completing the checkpoints during VR test.

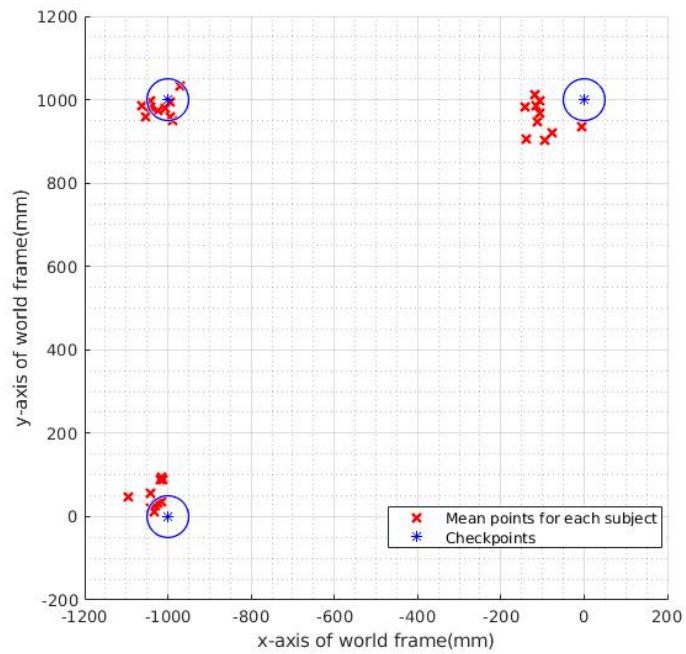


Figure 4.18. Mean position of the robot while completing the checkpoints during AR test.

direction of the movement. The first checkpoint is the second easiest since no obstacle is present.

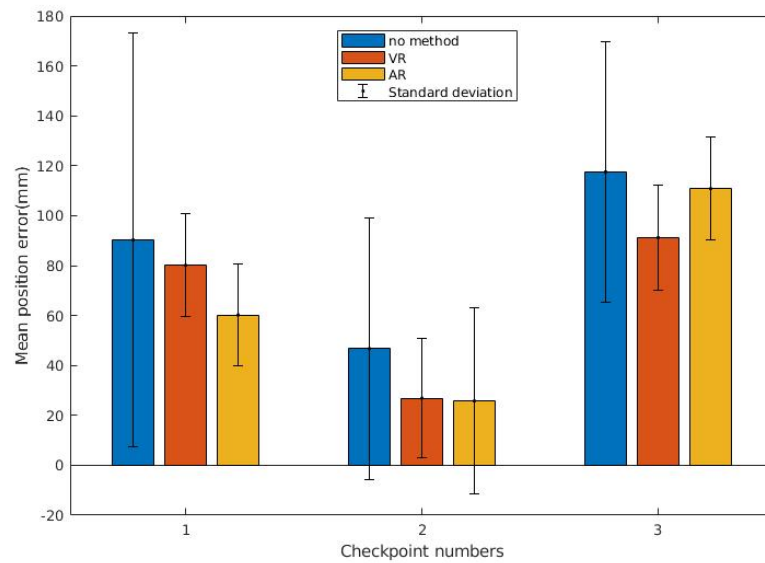


Figure 4.19. Mean position of the robot while completing the checkpoints during all tests.

The third checkpoint would appear to be the hardest. Because of the side obstacle, the resultant force feedback is in the diagonal direction which makes it harder to stay inside the checkpoint area.

Standard deviation values give an approximation about how much the subjects moved during each task. In standard teleoperation method, the deviation is much higher compared to other methods. Additionally, mean position is also higher in this method. It is concluded that model mediated approach improves the performance of the teleoperation.

4.3.3. User Questionnaire Results

User feedback results are presented in Table 4.3. The answers of the subjects are visualized in a compact form by utilizing numbers and colors for each teleoperation method. The representation is as follows: 1 and red for the delayed teleoperation method, 2 and blue for the VR method, 3 and green for the proposed AR method.

For the first question, VR method was chosen as the easiest to use by 8 out of 10 subjects. The reason for this is that the subjects can only see the virtual robot and not

Table 4.3. User questionnaire results (1: No method, 2: VR method, 3: AR method).

Questions	Subjects									
	1	2	3	4	5	6	7	8	9	10
Ease of use	2	2	2	2	2	2	3	3	2	2
Safety	3	3	3	1	3	3	3	3	3	3
Sense of task completion	2	2	1	3	3	3	3	3	2	3
Telepresence	3	3	3	3	3	3	3	3	2	3
Preference	3	3	3	3	3	3	3	3	3	3

overloaded by the extra sensory information as in AR method. Therefore, they can focus on reaching the checkpoints more easily.

Second question was about the safest method to use in teleoperation. 9 out of 10 subjects chose the AR method over other methods. Being able to interact with the environment without any delay and to see the real robot in the real environment proved to be the safest method.

The sense of task completion means that the users actually felt like accomplishing the tasks given to them. 6 of 10 subjects picked the AR method because they were both provided with the color indicator showing them the progress of the tasks as well as seeing the real robot completing the tasks.

After the subjects were given a brief description of telepresence, they were asked to select the method that provided the highest telepresence for them. 9 out of 10 subjects opted for the AR method, proving that the proposed method has increased the performance of the teleoperation compared to standard model-mediated teleoperation method.

As the answer to the last question, all of the subjects preferred using the augmented reality based model-mediated teleoperation method for similar teleoperation scenarios.

According to user test results, the performance of VR and AR based methods were equal and higher than standard teleoperation method. Taking all feedback into account including the feedback from the users, the proposed method was proven to increase the overall performance of the teleoperation by providing higher telepresence.

CHAPTER 5

CONCLUSION

Within the scope of this study, a teleoperation architecture based on model mediated teleoperation method was designed and implemented. An omni-directional mobile slave robot suitable for bilateral teleoperation was designed and built including the software and the hardware. A test setup to validate the proposed method was prepared and investigated on ten test subject.

User tests were conducted for three different teleoperation methods under 2 s round-trip communication delay: Standard bilateral teleoperation without any methods to reduce the effects of communication delay, standard model mediated teleoperation method and augmented reality based model mediated teleoperation method. The test results were evaluated for each subject and for each task separately according to total completion time, accuracy and precision of subjects and tasks, and feedback from the subjects.

In this study, a novel augmented reality based model-mediated teleoperation method was proposed and applied to an unlimited workspace application. Delayed visual feedback received from the remote environment is enhanced utilizing augmented reality.

According to the subjects, being able to see both the master and the slave in the same image in the real environment is preferred over the virtual reality version of the model-mediated teleoperation method. Even though the proposed method is not chosen to be the easiest to use by majority of the test subjects, telepresence and overall performance of the teleoperation is improved.

As a future work for the current test setup, accuracy of the pose estimation algorithm may be improved by utilizing an inertial measurement unit and/or encoder data for odometry calculations. Computational cost of the system may be reduced by optimizing the image processing so that the resolution of video feedback can be increased. Instead of a constant 1 second communication delay in each direction, variable time delays can be studied. Additionally, package loss in communication line may be introduced to better reflect real-world conditions such as teleoperation over internet.

This thesis is a proof of concept work for the project “Robotic Squid for Underwater Manipulation and Intervention”, funded by *The Scientific and Technological Research Council of Turkey (TÜBİTAK)*. The proposed method and the teleoperation architecture

will be applied on the underwater vehicle to be used in exploration and manipulation missions. However, various modifications are required to be applied to implement the proposed method for the underwater vehicle. Since the method was developed and tested in two dimensional workspace and the workspace of the underwater robot is three dimensional, new sensors suitable for this operation need to be deployed. In order to construct the virtual environment, an underwater LIDAR and a sonar sensor will be used. Force generation algorithm will also be modified to be used with point-cloud data utilizing two haptic devices instead of one. Lastly, augmented reality feedback will be changed from third-person to first-person perspective.

REFERENCES

- Anderson, R. and M. Spong (1989, May). Bilateral control of teleoperators with time delay. *IEEE Transactions on Automatic Control* 34(5), 494–501.
- Aselsan. Aselsan — kaplan unmanned ground vehicle family. <https://www.aselsan.com.tr/en/capabilities/unmanned-systems/unmanned-vehicles/kaplan-unmanned-ground-vehicle-family/>. (Accessed on 08/03/2019).
- DaVinci. Da vinci surgery — robotic assisted surgery for patients. <https://www.davincisurgery.com/>. (Accessed on 08/03/2019).
- Eca. Eca group — h200-sur / rov / remotely operated vehicle. <https://www.ecagroup.com/en/solutions/h2000-sur-rov-remotely-operated-vehicle/>. (Accessed on 08/03/2019).
- Farkhatdinov, I. and J.-H. Ryu (2007). Hybrid position-position and position-speed command strategy for the bilateral teleoperation of a mobile robot. In *2007 International Conference on Control, Automation and Systems*, pp. 2442–2447. IEEE.
- Ferrell, W. R. (1965). Remote manipulation with transmission delay. *IEEE Transactions on Human Factors in Electronics* (1), 24–32.
- Ferrell, W. R. and T. B. Sheridan (1967). Supervisory control of remote manipulation. *IEEE spectrum* 4(10), 81–88.
- Franklin, G. F., J. D. Powell, M. L. Workman, et al. (1998). *Digital control of dynamic systems*, Volume 3. Addison-wesley Menlo Park, CA.
- Hokayem, P. F. and M. W. Spong (2006). Bilateral teleoperation: An historical survey. *Automatica* 42(12), 2035–2057.
- Hunt, K. H. and F. R. E. Crossley (1975). Coefficient of restitution interpreted as damping in vibroimpact. *Journal of applied mechanics* 42(2), 440–445.
- Kohlbrecher, S., O. von Stryk, J. Meyer, and U. Klingauf (2011, November). A flexible and scalable SLAM system with full 3d motion estimation. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*. IEEE.
- Lawrence, D. A. (1993). Stability and transparency in bilateral teleoperation. *IEEE transactions on robotics and automation* 9(5), 624–637.
- Liu, C., J. Guo, and P. Poignet (2018). Nonlinear model-mediated teleoperation for surgical applications under time variant communication delay. *IFAC-PapersOnLine* 51(22), 493–499.
- Mitra, P. and G. Niemeyer (2008a). Mediating time delayed teleoperation with user

- suggested models: Implications and comparative study. In *2008 Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 343–350. IEEE.
- Mitra, P. and G. Niemeyer (2008b). Model-mediated telemanipulation. *The International Journal of Robotics Research* 27(2), 253–262.
- NASA. Restore-1 — satellite servicing projects division, robotics servicing mission. <https://sspd.gsfc.nasa.gov/restore-L.html/>. (Accessed on 08/03/2019).
- Niemeyer, G. D. (1996). *Using wave variables in time delayed force reflecting teleoperation*. Ph. D. thesis, Massachusetts Institute of Technology.
- Panzirsch, M., H. Singh, M. Stelzer, M. J. Schuster, C. Ott, and M. Ferre (2018). Extended predictive model-mediated teleoperation of mobile robots through multi-lateral control. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1723–1730. IEEE.
- Passenberg, C., A. Peer, and M. Buss (2010). Model-mediated teleoperation for multi-operator multi-robot systems. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4263–4268. IEEE.
- Rowekamper, J., C. Sprunk, G. D. Tipaldi, C. Stachniss, P. Pfaff, and W. Burgard (2012, October). On the position accuracy of mobile robot localization based on particle filters combined with scan matching. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE.
- Ryden, F. and H. J. Chizeck (2013). A proxy method for real-time 3-dof haptic rendering of streaming point cloud data. *IEEE transactions on Haptics* 6(3), 257–267.
- Ryden, F., S. N. Kosari, and H. J. Chizeck (2011). Proxy method for fast haptic rendering from time varying point clouds. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2614–2619. IEEE.
- Sheridan, T. B. (1989). Telerobotics. *Automatica* 25(4), 487–507.
- Stakem, P. (2014). Robots and telerobots in space applications.
- Stassen, H. G. and G. J. Smets (1995, June). Telemanipulation and telepresence. *IFAC Proceedings Volumes* 28(15), 13–23.
- Sturm, P. (2014). Pinhole camera model. In *Computer Vision*, pp. 610–613. Springer US.
- Taner, B., M. İ. C. Dede, and E. Uzunoğlu (2015). Applying model mediation method to a mobile robot bilateral teleoperation system experiencing time delays in communication. *Makina Teorisi Derneği*.
- Trevelyan, J. P., S.-C. Kang, and W. R. Hamel (2008). Robotics in hazardous applica-

tions. *Springer handbook of robotics*, 1101–1126.

Uzunoglu, E. and M. İ. C. Dede (2017). Extending model-mediation method to multi-degree-of-freedom teleoperation systems experiencing time delays in communication. *Robotica* 35(5), 1121–1136.

Valenzuela-Urrutia, D., R. Muñoz-Riffo, and J. Ruiz-del Solar (2019, Feb). Virtual reality-based time-delayed haptic teleoperation using point cloud data. *Journal of Intelligent & Robotic Systems*.

Willaert, B., J. Bohg, H. Van Brussel, and G. Niemeyer (2012). Towards multi-dof model mediated teleoperation: using vision to augment feedback. In *2012 IEEE International Workshop on Haptic Audio Visual Environments and Games (HAVE 2012) Proceedings*, pp. 25–31. IEEE.

Xu, X., B. Cizmeci, and E. G. Steinbach (2013). Point-cloud-based model-mediated teleoperation. In *HAVE*, pp. 69–74.

Yazdankhoo, B. and B. Beigzadeh (2019). Increasing stability in model-mediated teleoperation approach by reducing model jump effect. *Scientia Iranica* 26(1), 3–14.

APPENDIX A

TEST DATA

A.1. Motor Controller Test

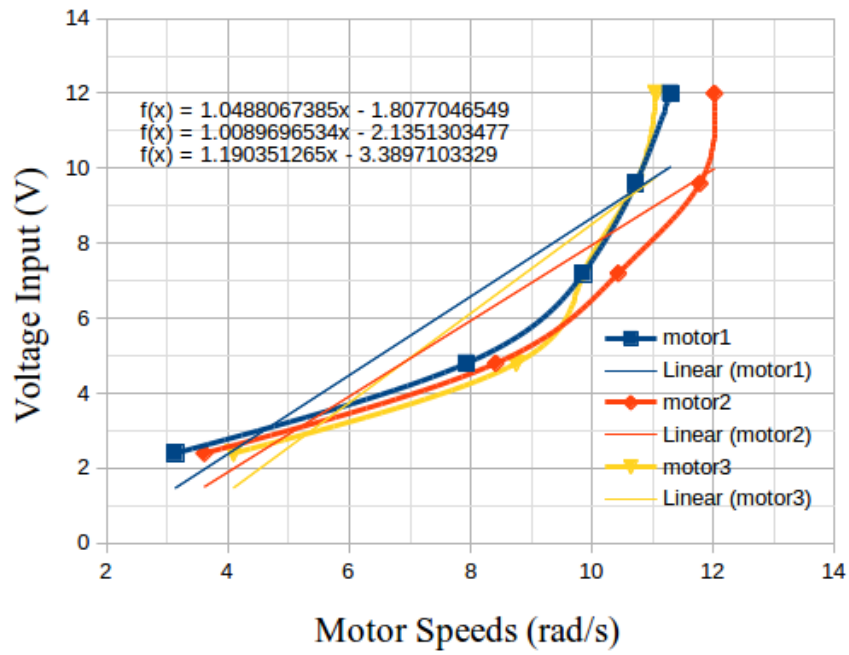


Figure A.1. Motor speeds vs voltage input to the motors.

A.2. User Test Data

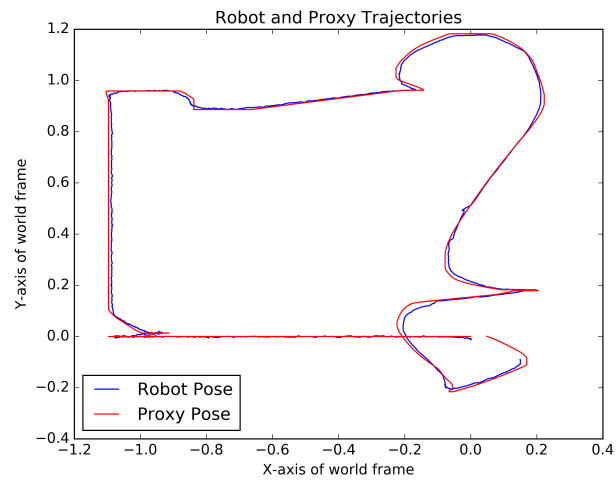


Figure A.2. Trajectory of direct bilateral teleoperation for User 1.

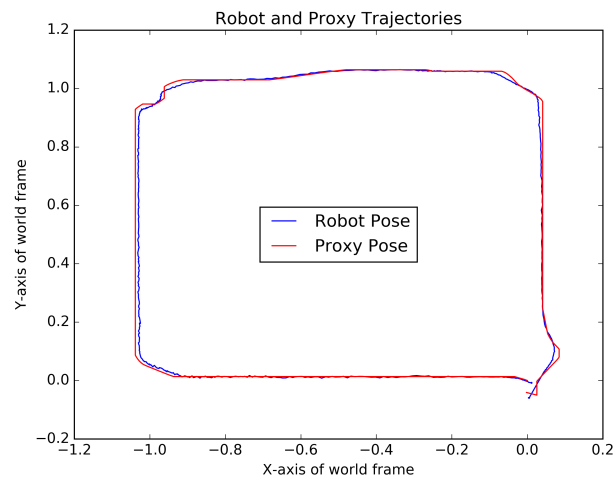


Figure A.3. Trajectory of virtual reality-based MMT for User 1.

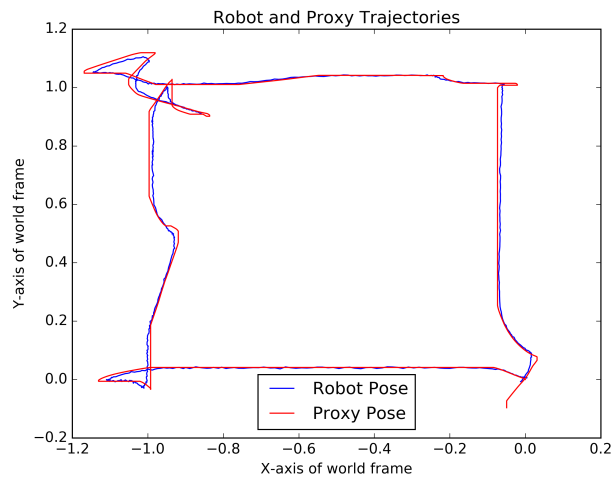


Figure A.4. Trajectory of augmented reality-based MMT for User 1.

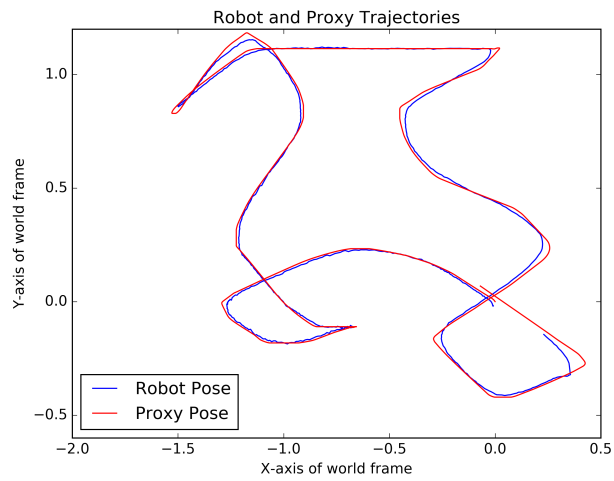


Figure A.5. Trajectory of direct bilateral teleoperation for User 2.

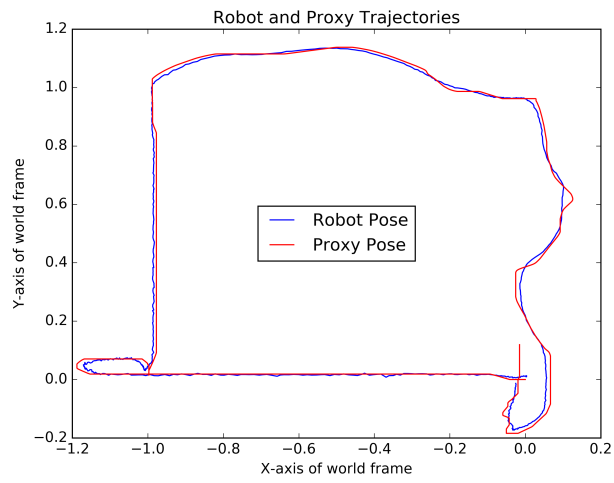


Figure A.6. Trajectory of virtual reality-based MMT for User 2.

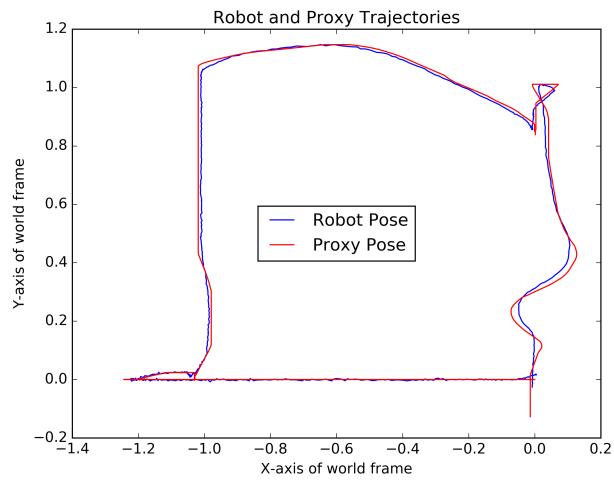


Figure A.7. Trajectory of augmented reality-based MMT for User 2.

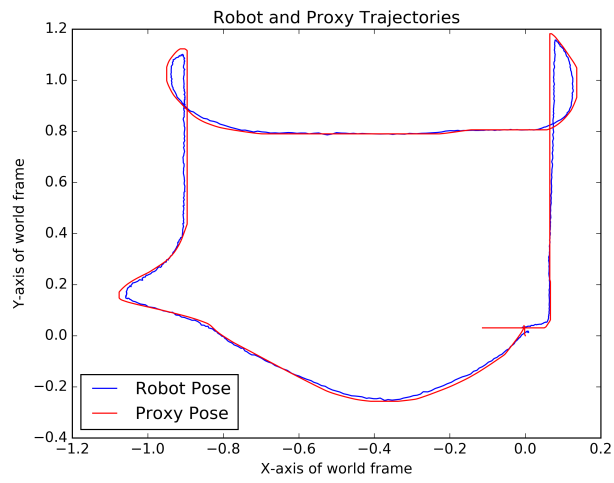


Figure A.8. Trajectory of direct bilateral teleoperation for User 3.

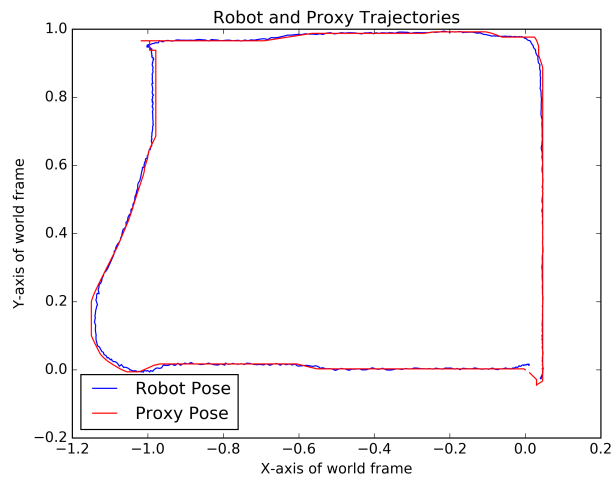


Figure A.9. Trajectory of virtual reality-based MMT for User 3.

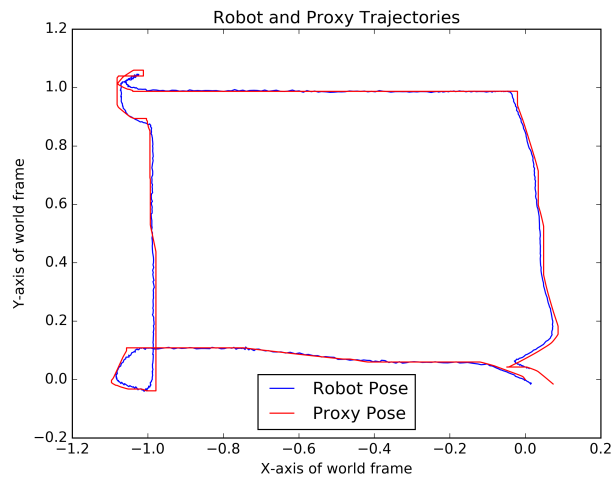


Figure A.10. Trajectory of augmented reality-based MMT for User 3.

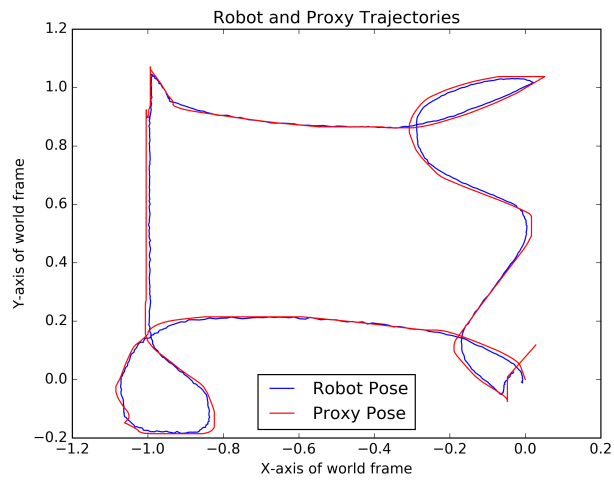


Figure A.11. Trajectory of direct bilateral teleoperation for User 4.

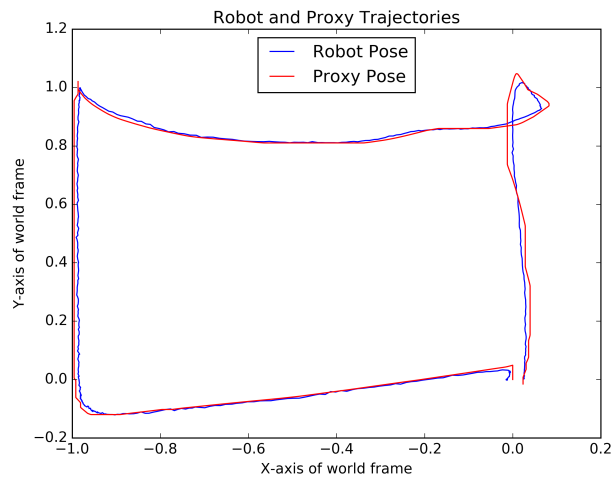


Figure A.12. Trajectory of virtual reality-based MMT for User 4.

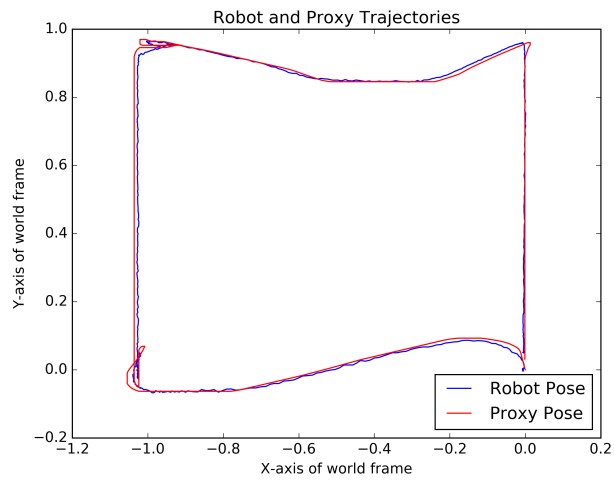


Figure A.13. Trajectory of augmented reality-based MMT for User 4.

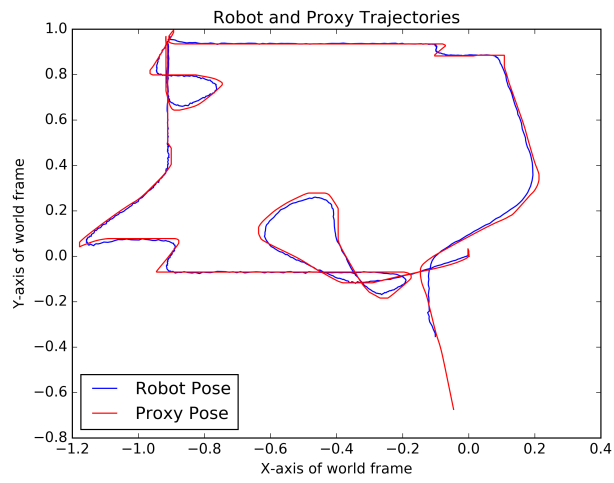


Figure A.14. Trajectory of direct bilateral teleoperation for User 5.

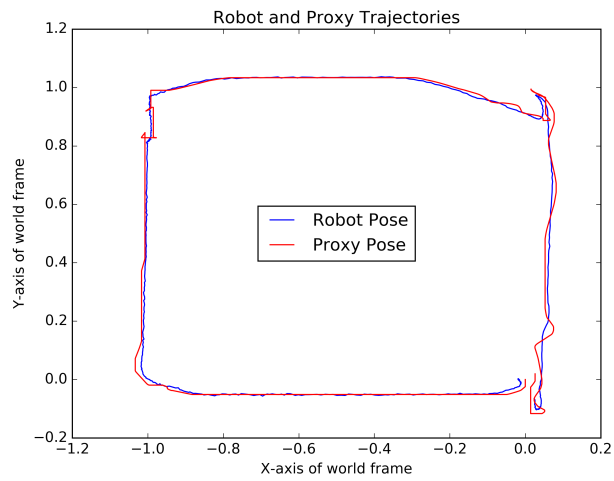


Figure A.15. Trajectory of virtual reality-based MMT for User 5.

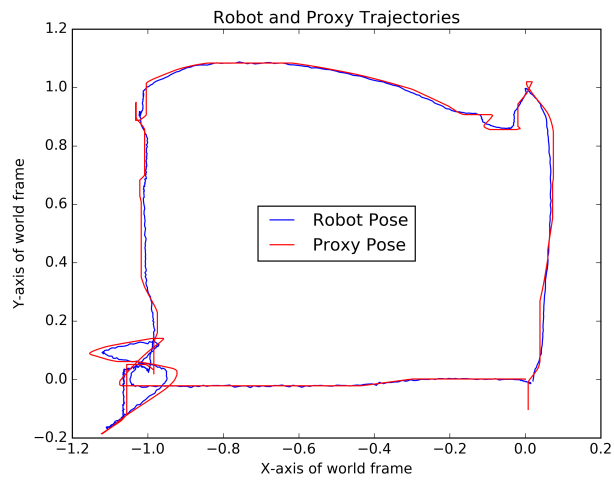


Figure A.16. Trajectory of augmented reality-based MMT for User 5.

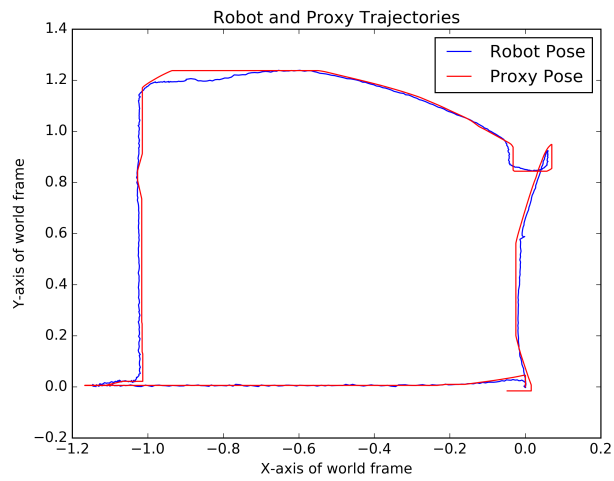


Figure A.17. Trajectory of direct bilateral teleoperation for User 6.

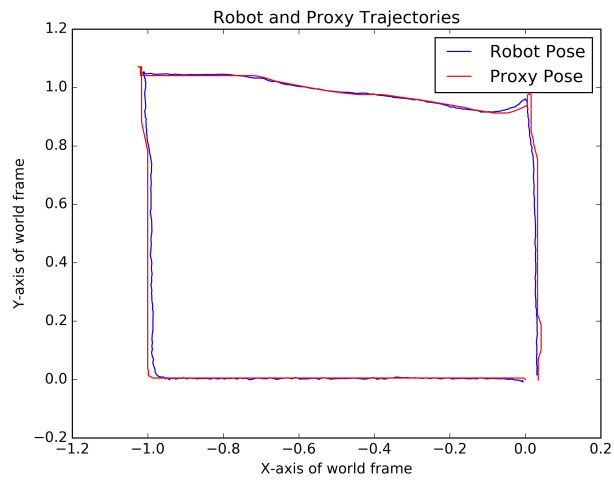


Figure A.18. Trajectory of virtual reality-based MMT for User 6.

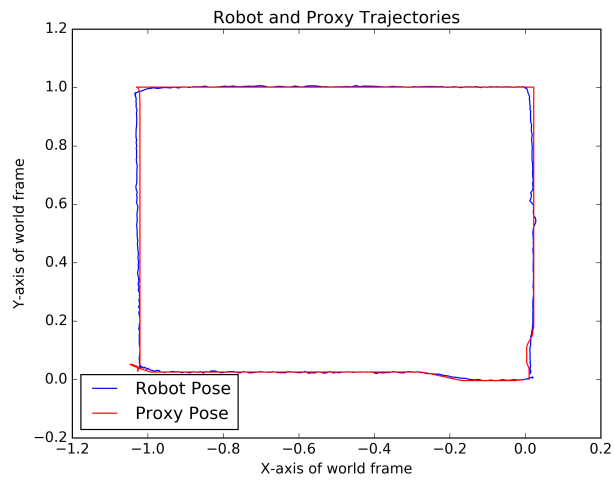


Figure A.19. Trajectory of augmented reality-based MMT for User 6.

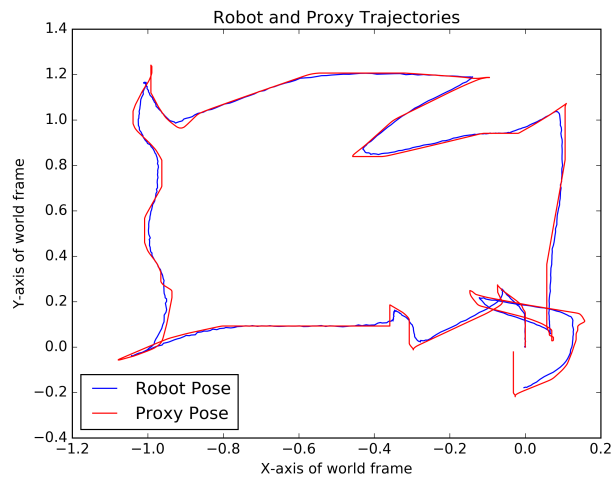


Figure A.20. Trajectory of direct bilateral teleoperation for User 2.

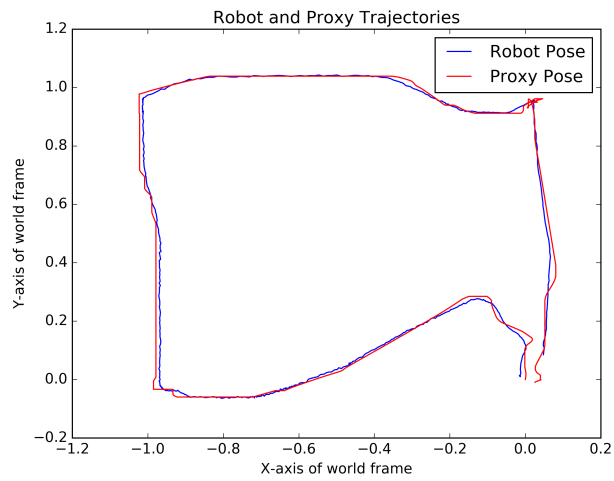


Figure A.21. Trajectory of virtual reality-based MMT for User 7.

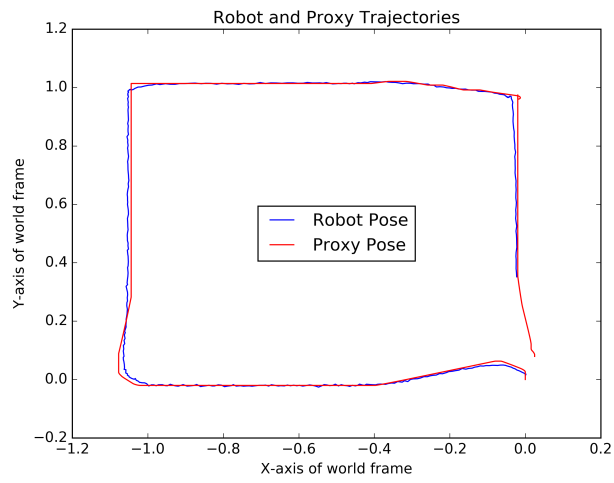


Figure A.22. Trajectory of augmented reality-based MMT for User 7.

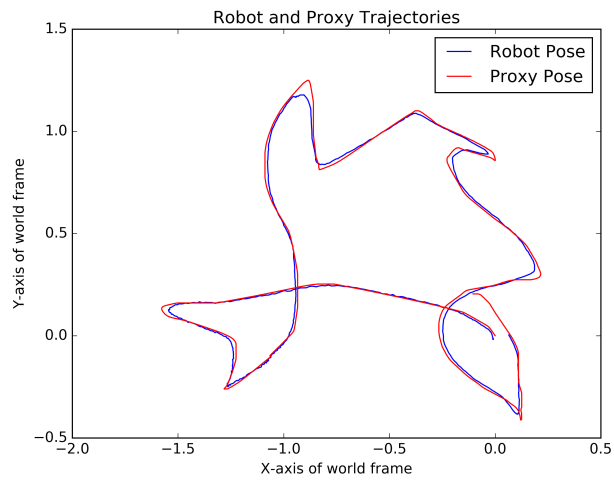


Figure A.23. Trajectory of direct bilateral teleoperation for User 8.

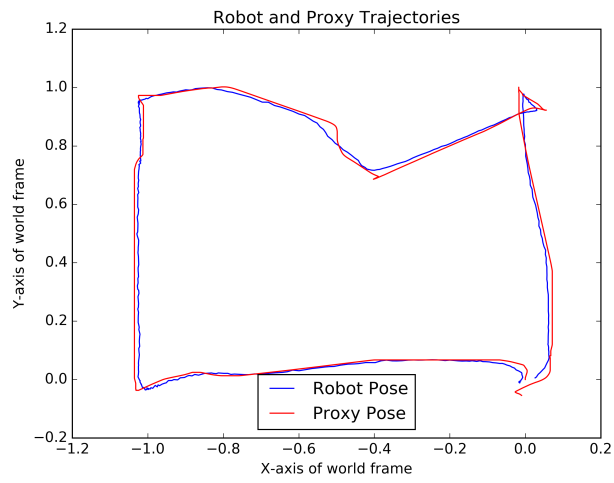


Figure A.24. Trajectory of virtual reality-based MMT for User 8.

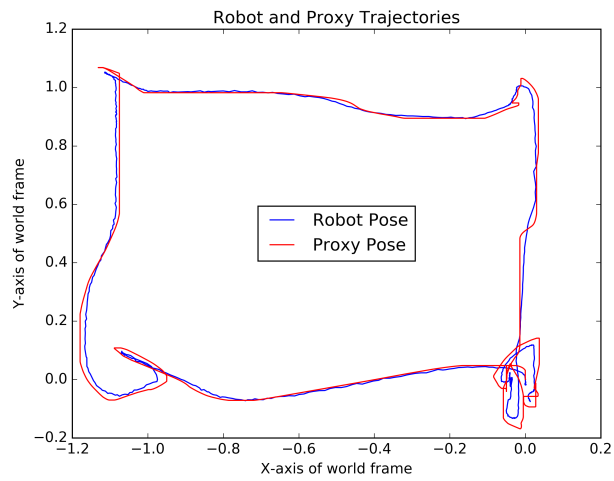


Figure A.25. Trajectory of augmented reality-based MMT for User 8.

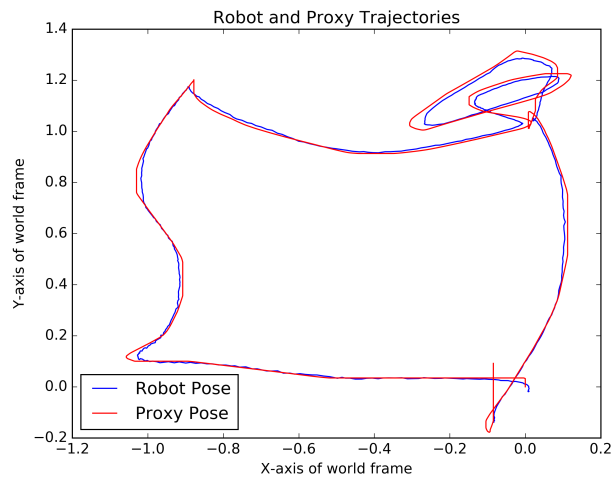


Figure A.26. Trajectory of direct bilateral teleoperation for User 9.

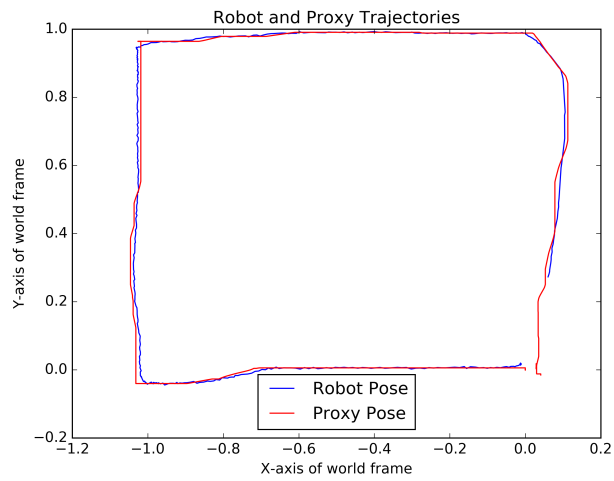


Figure A.27. Trajectory of virtual reality-based MMT for User 9.

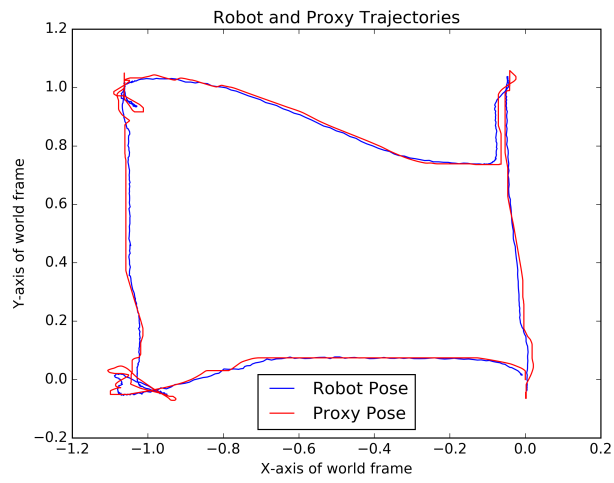


Figure A.28. Trajectory of augmented reality-based MMT for User 9.

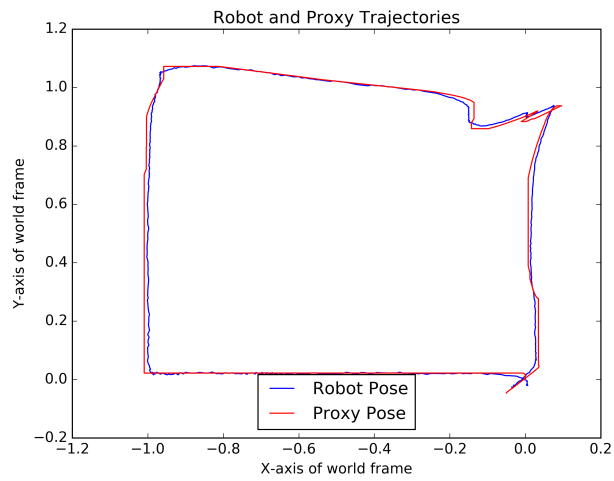


Figure A.29. Trajectory of direct bilateral teleoperation for User 10.

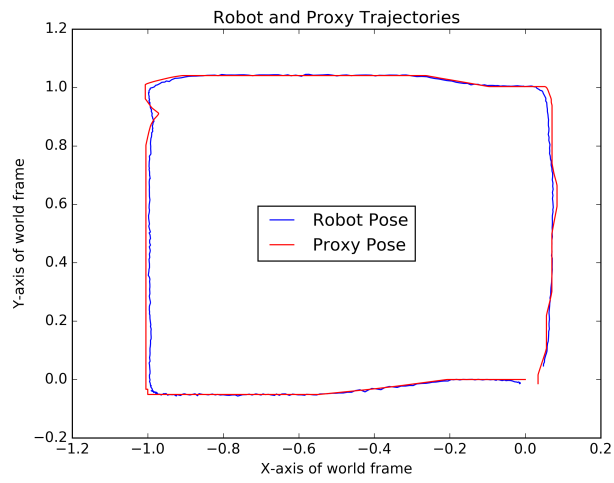


Figure A.30. Trajectory of virtual reality-based MMT for User 10.

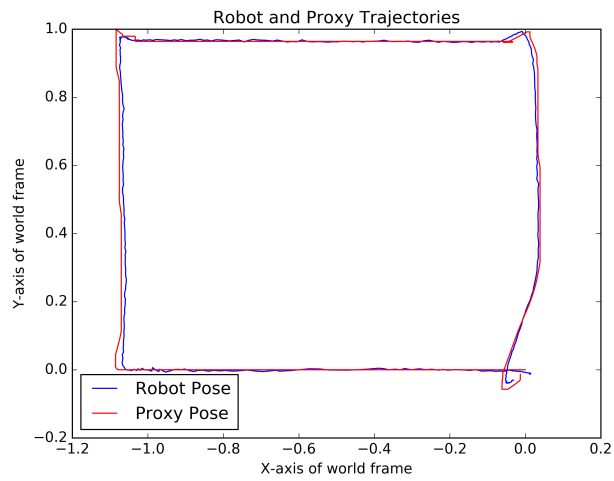


Figure A.31. Trajectory of augmented reality-based MMT for User 10.