KTH Computer Science
and Communication

Input Estimation for Teleoperation

Using minimum jerk human motion models to improve telerobotic performance

CHRISTIAN SMITH

Doctoral Thesis
Stockholm, Sweden 2009

TRITA-CSC-A 2009:20
ISSN 1653-5723
ISRN KTH/CSC/A–09/20-SE
ISBN 978-91-7415-517-4

KTH School of Computer Science and Communication
SE-100 44 Stockholm
SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av Teknologie doktorsexamen i datalogi fredagen den 11 december 2009 klockan 13.00 i F2, Kungl Tekniska högskolan, Lindstedtsvägen 28, Stockholm.

Tryck: Universitetsservice US AB

Abstract

This thesis treats the subject of applying human motion models to create estimators for the input signals of human operators controlling a telerobotic system.

In telerobotic systems, the control signal input by the operator is often treated as a known quantity. However, there are instances where this is not the case. For example, a well-studied problem is teleoperation under time delay, where the robot at the remote site does not have access to current operator input due to time delays in the communication channel. Another is where the hardware sensors in the input device have low accuracy. Both these cases are studied in this thesis.

A solution to these types of problems is to apply an estimator to the input signal. There exist several models that describe human hand motion, and these can be used to create a model-based estimator. In the present work, we propose the use of the minimum jerk (MJ) model. This choice of model is based mainly on the simplicity of the MJ model, which can be described as a fifth degree polynomial in the cartesian space of the position of the subject's hand.

Estimators incorporating the MJ model are implemented and inserted into control systems for a teleoperated robot arm. We perform experiments where we show that these estimators can be used for predictors increasing task performance in the presence of time delays. We also show how similar estimators can be used to implement direct position control using a handheld device equipped only with accelerometers.

Sammanfattning

Denna avhandling beskriver hur man kan tillämpa modeller for mänsklig rörelse för att skapa estimatorer för styrsignalerna en mänsklig operatör ger ett fjärrstyrt robotsystem.

Man betraktar ofta operatörens indata som en känd storhet i fjärrstyrda robotsystem. Det finns dock tillfällen när denna beskrivning inte är tillämpbar. Ett välkänt exempel är när man har tidsfördröjd fjärrstyrning, så att den fjärrstyrda roboten inte har tillgång till operatörens nuvarande indata. Ett annat exempel är när mätutrustningen i användargränssnittet har begränsad noggrannhet. Båda dessa fall avhandlas i denna text.

En lösning för den här typen av problem är att använda en estimator för insignalen. Det finns modeller som beskriver mänskliga handrörelser, och dessa kan användas för att skapa en modellbaserad estimator. I den här avhandlingen föreslås den s.k. *minimum jerk*-modellen (MJ). Valet av modell baseras främst på modellens enkelhet; modellen kan uttryckas som ett femtegradspolynom i kartesiska koordinater för handens position.

Estimatorer som bygger på MJ-modellen implementeras och tillämpas i styrsytemet för en fjärrstyrd robotarm. Vi utför experiment där vi visar att dessa estimatorer kan användas för att förbättra prestanda för fjärrstyrning med tidsfördröjd kommunikation. Vi visar också hur liknande estimatorer kan användas för att implementera direkt positionsstyrning med en handhållen apparat enbart försedd med accelerometrar.

Acknowledgments

The thesis you are presently reading is a report summarizing my research performed as a graduate student at CAS, the Centre for Autonomous Systems at The Royal Institute of Technology. As is almost always the case with these things, even though I accept full responsibility for the final product, there are of course several people who have influenced both the process and the product, and I would like to use this space for some sincere thanks.

In chronological order, these thanks start with Henrik Christensen, my original advisor and the person who provided me with everything necessary for meaningful research: problems to work on, funding, feedback, and freedom to proceed in the direction I wanted. I'm grateful for the opportunities for travelling and meeting other practitioners in the field and seeing their work. I would also like to express my gratitude that you stayed on as my advisor long after moving to Georgia Tech in Atlanta.

I would like to thank Patric Jensfelt for accepting to become my advisor for the final year. I understand that it must have been difficult entering into the research project at such a late stage, but nonetheless I have received more enthusiasm and constructive feedback than I could ever had hoped for.

I thank Mattias Bratt for being a good collaborator and colleague. We built some interesting things together and I, at least, had a lot of fun moments. The work on teleoperated ballcatching presented in this thesis would not have been possible without your collaboration. I would also like to thank Danica for originally introducing me to the lab and this field of research. It is my hope that this thesis will be accepted so that the next time you introduce me as Dr. Smith, the title will actually be correct. I thank Josephine and Stefan for making an effort to make feel a part of the lab after my project was orphaned with only me left working on it, and JOE for sharing your vast experience of things academic.

I also want to mention my roommates, who have come and gone over the years: Arvid, John, Per, and Magnus. I hope that you have been able to get some meaningful work done despite my constant interruptions, and that you have enjoyed my company as much as I have yours. As I promised, I hereby especially mention that Per was the originator of the idea to use floorballs for one of the experimental setups. My thanks to the other grad students at CAS: Elin for all the interesting discussions on parenting and life in general, Oscar for all the lunches when you have endured my ranting on the topic of the day, Carl for providing fresh observations on the absurdities that emerge when cultures clash, Staffan, Daniel, Paul, Hugo, and Frank for enforcing the 3 o'clock coffee breaks, Babak, Niklas, Jeanette, Alper, Kristoffer, Javier, Andrzej, and Oskar for being willing guinea pigs in my experiments (as have most people in the lab at one time or another, come to think of it). Also, thanks to Sagar for wanting to use the robot for your master thesis, forcing me to produce a comprehensible API.

Furthermore, my thanks go to our administrative staff: Linda, Mariann, Jeanna, and Friné. I never seem able to learn the procedures for paperwork, and I really appreciate the patience with which you have helped me through the process time and time again.

More than anything, however, I am eternally grateful for all the sacrifices made and support given by my beloved wife Kaori. Without you, this thesis would never have seen the light of day.

Contents

Chapter 1

Introduction

This thesis is about controlling robots via teleoperation. While a relatively young field, tracing its origins to the middle of the last century, teleoperation has already been used in a large variety of situations. Teleoperated robots have handled volatile matter, from radioactive fuels to hostage situations. They have been used to explore both the depths of the ocean and the vastness of space. They range in scale and payload from precision devices like the "Da Vinci" surgical robot to the 15 meter "Canadarm" robot handling payloads up to 10 tons on the Space Shuttle.

1.1 The Human in Control

Why then, are teleoperated robots of interest? Upon hearing the word *robot*, the first thing that would come to mind for most is probably an image of an autonomous device — an industrial robot performing repetitive tasks significantly faster and with higher precision than a human worker, or a free moving device like a space exploring rover or an automated vacuum cleaner, or perhaps even more exotic machines like humanoid robots rarely found outside the realms of research laboratories or works of fiction.

However, with teleoperation, the robot is controlled by a human operator. There may be several reasons for this, but most can be summarized as a need for the human's superior skills of interpretation and analysis of the present situation and ability to react and adapt to unexpected events and changes in the environment. There exists a multitude of tasks that humans excel at, but which machines are not yet able to perform unsupervised. These tasks range from those requiring trained expert skills, like advanced surgery or space station repairs, to those that require human interaction skills, like hostage negotiation, or just need human innovation and adaptivity to novel conditions when exploring unknown environments.

The machine can nonetheless be used to extend the humans capabilities in several different ways, symbiotically utilizing the separate advantages of man and machine. Machines can be present where it would be too dangerous or too expensive to send a human — in outer space, at the bottom of the sea, or in nuclear reactors to name a few examples. Also, machines may have different scales than their human operators, and enable these to handle very heavy objects, or perform surgery on a scale much too small for human fingers, while the human provides the skills, knowledge and analysis necessary for the task.

1.2 Problems Addressed

This thesis focuses on problems that arrise when measuring the input from the human operator in the presence of different types of measurement uncertainties. One of the main problems faced when connecting the human operator as a control device for a robotic system is that the direct control signal in the operator's brain is not directly available. Thus, different indirect means of estimating this have to be employed.

Some exploratory work has been done examining the possibilities of directly connecting a human brain to a machine interface, but this is still rudimentary and require invasive surgery to connect to the nervous system, and a considerable degree of noise still remains in the connection. Though this may be acceptable for some prosthetic use, easy-to-use systems for teleoperation are not expected to be accessible in the overseeable future [18, 106].

In present teleoperation setups, more indirect connections are usually employed. Most typically, information about the machine is presented to the operator via his/her physical senses, either by direct observation of the machine, or via some type of display. Visual, audio, and haptic modalities are the most common. The operator's control signal is then input to the system via some input device, which in essence consists of one or several sensors that measure conscious movements of some part of the operator. A typical mundane example may be a joystick that measures hand movements.

As with all sensor-based systems, there exists a certain degree of uncertainty in the mapping between the underlying process and the measured quantities, here represented by the operators intentions for the machine system and the sensor signals from the input device. Even if we were to assume that the operator conveys their desired control signal to the input device without error, the signal from the input device to the robot may be corrupted by transmission delays or sensor noise, adding further uncertainty to the original signal.

The main problem addressed in the present thesis is how treat this uncertainty. More specifically, the thesis investigates how a model of typical conscious human motion can be used to construct an estimator for the human control input signal, and how such an estimator then can be used to generate a better estimate of the human operator's desired control signal than the original sensor measurements from the input device of the user interface.

1.3 Contributions

The main contribution of this work is the proposition to use human motion models for estimating the input from the operator, and the demonstration of implementations of model-based estimators. By introducing a predictor for input signals on the control signal side of a teleoperation setup, we extend the toolbox available for treating teleoperation with time delays. We also introduce simple motion models as a means to perform direct motion input using only very simple accelerometers.

There are also some minor contributions of this thesis, that are not yet completely verified but give us pointers to interesting further exploration. We show early results indicating that minimum jerk (MJ) motion models can be used as accurate descriptors of human operator input using devices ranging from a video game controller to a parallel link haptics interface.

The work presented here is the author's own work, but parts have been done in collaboration. The experiments described in Sections 6.1 and 7.1 were conducted in cooperation

with Mattias Bratt, who also implemented the 3D graphics user interfaces described in Section 5.2.

1.4 Disposition

The remainder of this thesis is disposed in the following way:

Chapter 2 – Background

This chapter introduces the concept of teleoperation and presents its history. Some problems that have arisen in different scenarios are discussed along with a brief description of how these have historically been approached. The current state of the art in teleoperation is summarized here.

Chapter 3 – Control Signal Modelling

In this chapter, we introduce the main idea of this thesis — using human motion models to construct an estimator for the human operator's input. Some different models describing human motion are presented and an argument is made in support of the minimum jerk (MJ) model. Related work using this and other models of human motion to estimate input is presented.

Chapter 4 – Experiment Design

Here, we propose what types of experiments to use for evaluating the validity of the models. We define minimal experiments to use for proof of concepts, along with more difficult experiments that can be used to establish the limits of what can be achieved. Experiments are discussed both in terms of what tasks to perform and in terms of what modes of interaction and which interfaces to employ.

Chapter 5 – Design of Experimental System

This chapter describes the setup used in the experiments. The setup includes a robot arm for the operator to control, several different user interfaces, both input devices and displays, and some external sensors used to measure reference positions. The communication structure of the teleoperation setup is also described. The design of the experimental setup itself has been presented in publications [21, 132, 133].

Chapter 6 – Offline Experiments

Here, we describe some exploratory experiments with human subjects. In these experiments, we record all data and use it afterwards for offline analysis. The MJ model is applied to this recorded data in different ways in order to find possible ways to implement MJ-based estimators. Parts of the results of these experiments have been presented in publications [22, 131, 135, 136].

Chapter 7 – Online Experiments

Using the results from the previous chapter, we implement MJ estimators that work in real-time, and perform experiments where the estimator output is used as an input signal

for robot control in different ways. Parts of the results of these experiments have been presented in publications [22, 131, 134, 135, 136].

Chapter 8 – Conclusions

In this final chapter, we summarize the thesis, the experimental results and present the conclusions to be drawn, along with comments on which results are conclusive, and which results that would need further inquiry.

1.5 Publications

Some of the results presented in this thesis have been previously published in the following papers:

[21] Mattias Bratt, Christian Smith, and Henrik I. Christensen. Design of a control strategy for teleoperation of a platform with significant dynamics. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1700–1705, Beijing, China, Oct 2006.

[22] Mattias Bratt, Christian Smith, and Henrik I. Christensen. Minimum jerk based prediction of user actions for a ball catching task. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2710–2716, San Diego, Ca, USA, Oct 2007.

[132] Christian Smith and Henrik I. Christensen. Using COTS to construct a high performance robot arm. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4056–4063, Rome, IT, April 2007. IEEE.

[131] Christian Smith, Mattias Bratt, and Henrik I Christensen. Teleoperation for a ballcatching task with significant dynamics. *Neural Networks, Special Issue on Robotics and Neuroscience*, 24, pages 604–620, May 2008.

[134] Christian Smith and Henrik I. Christensen. A minimum jerk predictor for teleoperation with variable time delay. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5621–5627, Saint Louis, USA, Oct 2009.

[135] Christian Smith and Henrik I. Christensen. Wiimote robot control using human motion models. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5509–5515, Saint Louis, USA, Oct 2009.

[133] Christian Smith and Henrik I. Christensen. Constructing a high performance robot from commercially available parts. (to appear) *Robotics and Automation Magazine*, vol. 16:4, Dec. 2009.

Parts of the results are currently under review for publication in the following paper:

[136] Christian Smith and Patric Jensfelt. A Predictor for Operator Input for Time-Delayed Teleoperation. *Mechatronics, Special Issue on Design Control Methodology*, 2010.

Chapter 2

Background

This chapter will give a brief introduction to the history of teleoperation, describe different types of telerobotic systems in use, discuss typical problems that can arise, and finally present some of the state of the art for dealing with these problems.

As the term itself implies, *teleoperation* is the process of performing some action at a distance. There does not necessarily exist a generally agreed upon clear definition of what is to be counted as teleoperation and what is not. Typical definitions are very wide and encompass a broad range of possible systems.

Sheridan defines *teleoperation* as "the extension of a person's sensing and manipulation capability to a remote location" [124].

Hokayem and Spong defines *teleoperation* as "Extend[ing] the human capability to manipulating objects remotely by providing the operator with similar conditions as those at the remote location" [68]

Rybarczyk defines *teleoperation* as "Indirectly acting on the world" [119]

For the present work, we shall use a broad definition, and let teleoperation mean the deployment of any system by which a human operator (the *master*) can control a piece of machinery (the *slave*) without being in direct contact. The distance that separates the master and slave is not significant for this definition, but the degree of separation should be such that regular tools with long handles are excluded, and we normally assume that the slave is actuated in some way.

2.1 History of teleoperation

Since the first tools were invented, it has been possible to transfer human action over space, time and scale. Different types of cranes, pincers, and tongs have allowed humans to manipulate objects that have been too large, too small, too far away or too dangerous to handle with bare hands.

However, the first instance of what could be thought of as a telerobotic system is probably the mechanical linkage manipulators used in the early 1950's to handle nuclear material. The systems were of the master-slave kind. A human user was operating a master input device and the slave manipulator reproduced the exact motion. Often the master and slave were coupled mechanically, so the teleoperation distance was not very far, but on the

other hand, the design inherently provided haptic feedback through the linkage, and visual feedback through a protective window [48, 151]. Eventually, these systems were replaced with manipulators with electrical servo motors, allowing for more separation of master and slave stations, but at the cost of feedback fidelity [29, 49]. These first systems were without exception only treating *teleoperated manipulation*. These systems employ *direct control*, meaning that the operator directly controls the slave, see Figure 2.1(a).

By the 1960's, the separation distance was increased as teleoperation was used for underwater applications. There were several different electric and electrohydralic manipulators used for manipulating objects at a depth of several thousand meters, with the operator located at the surface, or even on the shore [112]. At this time, in conjunction with the space race, both the United States and the Soviet Union developed teleoperated manipulators to be used for unmanned space exploration, with the *Surveyor* and *Lunakod* systems [76, 139]. Later, manned missions with the Space Shuttle were also aided by the 15 m "Canadarm" manipulator [2]. As the slave systems became mobile and were sent to places not accessible to the operator, *teleoperated locomotion* and *teleoperated sensing* also became important fields to study [104].

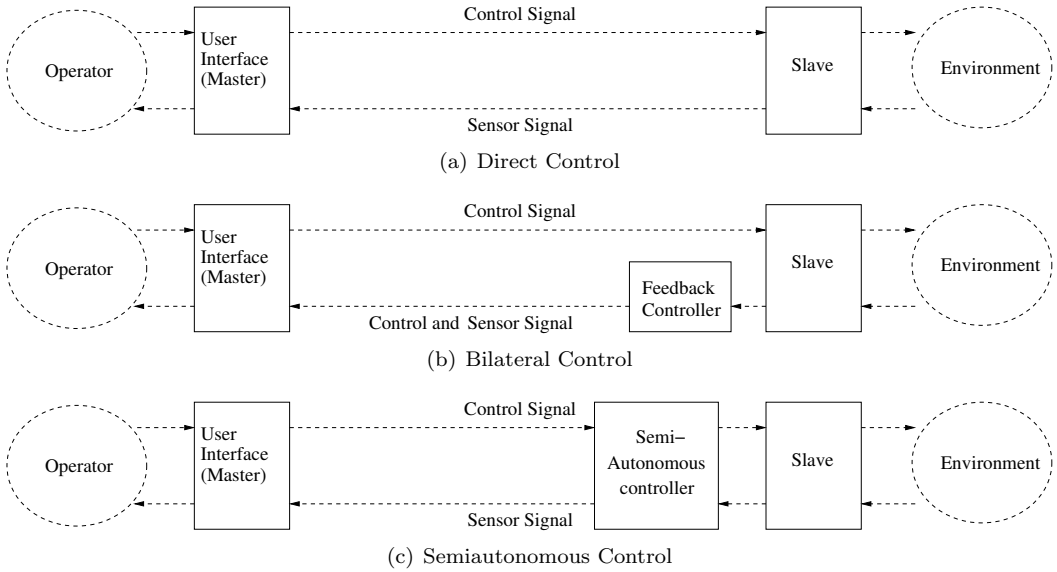
By 1972, a teleoperated robot known as the *wheelbarrow* was first employed to disarm bombs in the UK [16]. Since then, bomb disposal and ordnance clearing has become one of the most common uses of teleoperation, and teleoperated robots are standard equipment for many modern police and military bomb disposal units, where they are also sometimes put to use in other hazardous scenarios like hostage negotiation or surveillance of armed suspects [90].

Teleoperated robots were also introduced into surgery in the mid 1980's. In the beginning, the motivation was to provide stability for sensitive operations such as brain or heart surgery [114]. In the 1990's the application of different types of minimal invasive surgery increased, as robots made it possible to make surgical operations through minimal narrow openings, while increasing the number of available degrees of freedom [58, 45, 96]. While this type of robot surgery is typically performed with the surgeon in close vicinity to the patient and robot, using long distance communications to let a surgeon operate on a remote patient has also been realized, with the first successful intercontinental operation carried out on a pig in 1995 [118], and the first intercontinental operation on a human patient carried out in 2001 [97]. For these tasks, the sense of touch is important for performing well. Therefore, they often employ *bilateral control*, meaning that the slave is directly controlled by the human operator, but an automatic control loop generates a feedback signal that in turn directly controls the force feedback to the user interface, see Figure 2.1(b).

More recent work such as operation of the MARS rovers have transferred actions not only across space, but also across time, due to the significant time-delay of operating vehicles on another planet. The degree of autonomy is significant to enable efficient operation and to avoid direct closure of the control loop, which would be close to impossible [143, 15]. These systems are known as *semiautomatic control systems*. The role of the human operator is to supply the slave controller with setpoints, targets, or objectives to accomplish. The distribution of control between operator and robot can vary, from shared or cooperative control — where the robot for example may aid with keeping a prespecified distance or contolling some DoF's while the operator controls the rest — to supervisory control, where the operator chooses a task for the robot to perform autonomously [5], see Figure 2.1(c).

Prompted by the Kobe earthquake and the McVeigh bombing of a federal building in Oklahoma, teleoperated robots were introduced into urban search and rescue in the mid 1990's. The first actual deployment to help disaster victims was at the site of the collapsed World Trade Center buildings in New York in 2001 [34, 26].

Some of the most recent applications of teleoperation is using a humaoid robot to extend



(a) Direct Control

(b) Bilateral Control

(c) Semiautonomous Control

Figure 2.1: Teleoperation control types

as many of the operator's functions as possible to the remote site. The first examples include using a teleoperated robot to replace the human driver of heavy machinery at potentially dangerous earthquake sites [153]. More novel, exploratory work uses a human-like android robot to convey the presence of the operator to a remote site for human to human communication purposes [120, 110]. While most teleoperation scenarios emphasize *telepresence* as the operator's own sensation of being present at the remote site, these last cases emphasize the sensation the operator's presence as experienced by observers at the remote site.

Teleoperation Problems

The problems facing a successful implementation of teleoperation can roughly be divided into two types. The first type of problems concern how the operator and system interact. One has to decide on what modalities of the robot the operator should control. Here, the issue can be one of simplicity of operation versus versatility. For example, when teleoperating a humanoid robot with more than 30 degrees of freedom (DoF), it is simpler for the operator if he/she does not have to control the enrire robot directly, but the number of possible actions may be limited if some DoFs are not controllable [54]. Furthermore, depending on the task, different control spaces may give different performance [5]. Should the operator control all joints directly, as with a typical crane or backhoe, or should the end effector be controlled in cartesian space? In the chosen space, should the position be controlled directly, or should the operator control the velocity? Tasks with short precision motions benefit from the former, while tasks with longer, faster, motions normally benefit from the latter [102]. In some cases, as with a humanoid robot, there may exist intuitive mappings between human motor skills and the robot's capabilities, while these might be less obvious in other cases, such as three-armed snakelike surgical robots [129].

Related problems of the same type concern what information to display to the operator,

and how. Again there is a balance, where providing too much information feedback may cause sensory overload for the operator, while providing too little information may limit the operator's understanding of the remote site. Also, it is a nontrivial consideration as how to present sensory information from artificial senses that differ from the operator's human senses, such as radar scans, temperature mappings, velocity or inertia measurements and suchlike [124]. Depending on available bandwidth the amount of sensor data that is relayable may be severely limited [5]. Even with a multitude of available sensor data, most users may still focus on a single type, typically video feedback, and ignore other sensor readings [10].

The second type of problem concerns performance limitations. Differences of scale, available velocity and robot fragility may set unintuitive limitations on the robot that have to be conveyed to the operator [150, 53]. If the robot is not able to perform the commanded action, discrepancies between operator input and slave actions may introduce instabilities. One of the most significant, and consequently one of the most studied limitations is transmission time delays [64]. These can cause instability as the operator tries to correct for perceived errors that are due to delays that may be as small as one or a few tenths of a second [150, 38]. For the longer time delays of interplanetary teleoperation, direct control may be altogether impossible for all but the most trivial tasks [104, 61, 140]. Teleoperation with time delay is one of the main problems studied in the present work.

2.2 State of the Art

Since the problems associated with time-delayed teleoperation have been well-known for several decades, several techniques to deal with these problems have been proposed and applied [126, 6, 68]. In a short summary, the main approaches that have been used can be classified as one of the following:

- **Move-and-wait** was the first approach applied to time-delayed teleoperation. It consists of the user first executing a small part of the overall motion, and then waiting while the remote slave reacts to the command. The user then makes the next move, waits for the response and so on. It is robust and simple to implement. Performance, as measured by task completion time, degrades linearly with the size of the time delay [127, 38].

- **Task level/supervisory control** is mostly used when teleoperation bandwidth is very low, or delays are large. In most cases, task-level control lets the operator choose from a predefined set of tasks, that are then performed autonomously by the remote system. This is a common approach in space telerobotics due to the large distances and thereby long time-delays involved. In supervisory control, a controller at the slave site may perform low-level control like for example obstacle avoidance, while the human operator specifies the desired goal or trajectory. This approach requires either a competent autonomous system at the remote site, or good enough modelling so that all possible tasks can be pre-programmed [150, 152, 125, 51, 50, 19, 95, 20, 15, 104, 61, 140]. Similar to this is the use of virtual fixtures. In a peg-in-hole experiment that suffered 45% task execution time degradation under 450 ms delay, this was reduced to 3% using virtual fixtures [117]

- **Predictive/simulated displays** can be used when autonomous control at the remote site is difficult to achieve. This approach uses a simulation at the master side for real-time interaction by the operator. The remote site then performs the same motions. It is common to present the operator with both the simulation and the delayed

measurements from the remote site, so that the actual results can be observed. This approach requires a good model of the remote site in order to create a valid simulated environment [13, 85, 12, 65, 14, 83]. A variant is the "hidden robot" concept, where the remote robot is not displayed to the operator, but only the remote environment itself, which can be "felt" via haptic feedback to the user [82]. A special case of simulated display is teleprogramming, where the task is first performed in simulation until a satisfactory performance has been recorded. Only the satisfactory version is then replayed at remote site [57]. Recent model-mediated methods simulate the remote environment locally for high-bandwidth interaction, and use sensor data from the remote site to estimate and update model parameters [101].

- **Wave variables** are applicable for bilateral force feedback teleoperation. This basic idea of this approach is to use a coordinate rotation to transform velocity and force variables into wave variables that are sums and differences, respectively, of the original variables. One of the main strengths of this approach is the guaranteed stability via a passive communication channel, small sensitivity to model errors, and the proven ability to be adapted to variable delays [108, 105, 27, 60], while the major drawback is that the bandwidth is severely limited by the time delay, i.e., given a time delay τ, the performance for frequencies above $1/\tau$ is limited [59]. An attempt to circumvent this problem using a predictive model of the environment coupled with a passive communication layer was proposed in [43].

- **Predictive control** from classical closed-loop control theory, uses a prediction $\hat{y}$ of the measured state y to deal with delays. This has also been applied to teleoperation. This approach requires enough knowledge of the remote site to construct a valid predictor $\hat{y}$ [137, 25, 6, 123, 130, 111], and is applicable when the feedback signal consists of low-dimensional readily-modelled quantities like manipulator positions, velocities or forces.

Common for these approaches is that they do not provide any solution for when there is a need for fast, high-bandwidth interaction, and the feedback signal is difficult to model or predict, as for example would be the case with a video or audio feedback signal from an unmodelled remote environment.

Chapter 3

Control Signal Modelling

This chapter motivates why it is of interest to use human motion models in teleoperation, and how they can be applied. A brief overview of common models used to describe human motion is given, and arguments are presented in support of using the *minimum jerk* model. Finally, the details of the minimum jerk model are described, along with a brief presentation of how this model has been used in other work.

3.1 Model Based Input Estimation

In most approaches to teleoperation, the input from the user is considered a known, more or less error-free signal. For most cases this is a valid assumption, as the operator station is a controlled and well-known environment, and input devices impose no significant unknown errors into the system.

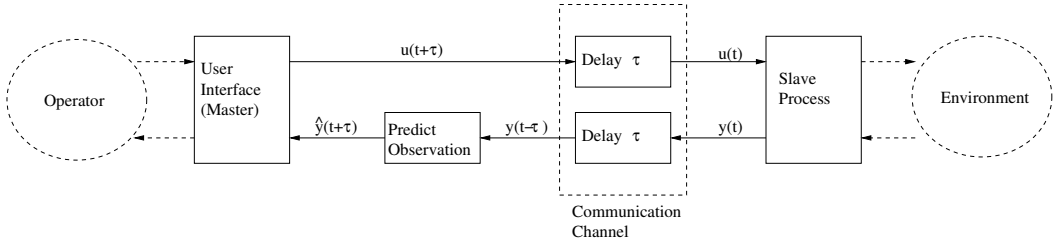
We can imagine cases where we even with highly controlled operator environments cannot observe the operator's input signal directly. The typical case is when we, for example due to long distances separating master and slave stations, have a considerable time delay. In this case, even with perfect measurements of the operator's input at the master station, we will only have access to a time-delayed version of this signal at the slave station.

One can also imagine delay-free scenarios where the assumption of low noise in the input signal starts to loose validity. The simplest scenario is when we have a limited budget or other limitations on the available input devices. This could include cases where the input device should be portable, or even incorporated as one function among many on a cheap portable device that can be used in any uncontrolled public environment, such as mobile phone, PDA, or a video game controller.

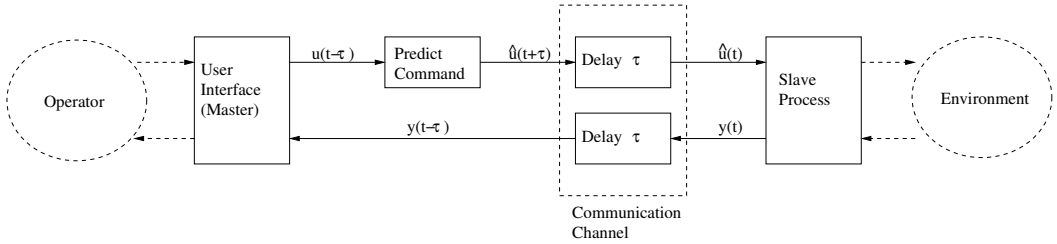
In the light of these two scenarios, the aim of the present work is to use models of human motion to compensate for imperfections in the input signal.

User Input Estimation and Prediction

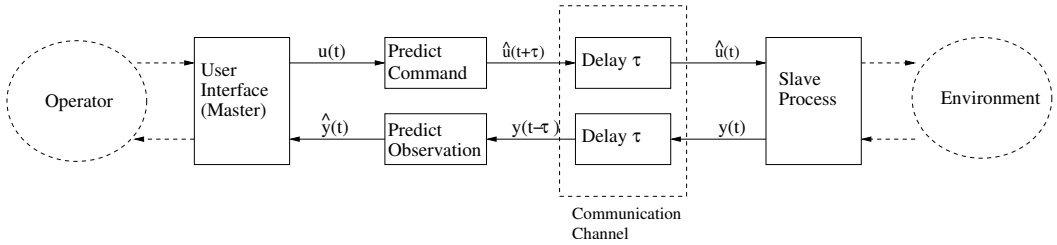
Estimation of stochastic or noisy signals is a well studied problem in the fields of signal processing and optimal filtering [77, 8, 9, 81, 7, 149]. Most methods require some model of the process that generates the signal that is estimated. The more accurate this model is, the better the estimator will perform. Thus, with a good model of human motion, it should be possible to use estimation methods from optimal filtering theory to fit this model to noisy observations.



(a) Teleoperation structure with prediction of observation



(b) Teleoperation structure with prediction of command signal



(c) Teleoperation structure with prediction of both observation and command signal

Figure 3.1: Teleoperation control structure with predictions to bridge time delay

As mentioned in the compilation of methods used to deal with time delays in teleoperation in the previous chapter, several methods include different ways to predict the remote site, or put in control terms, one substitutes the unavailable future measurement y with the predictor $\hat{y}$. A schematic of this is shown in Figure 3.1(a).

Given the possibility to use a model to predict the operator's command input signal, we can propose a novel control structure. Instead of handling the roundtrip delay by predicting the remote state with a simulation, the delay handling is moved to a command input predictor. The principal structure for this approach is shown in Figure 3.1(b). With this approach, measurements and video data from the remote site can be displayed as is, and there is no need for models of the remote site. Video feedback based control can therefore be performed with a camera with an unknown position in relation to the remote robot and task, as long as the camera shows an adequate view of the task space, enabling a human operator to interpret and react to the scene.

In cases where the process is highly non-linear, improvements can be made by not predicting the entire roundtrip delay at one instant, but predicting just the one-way delay for each of observation and command signals, as illustrated in Figure 3.1(c).

Limitations

Since one of the main motivations behind using direct control teleoperation as opposed to autonomous or semi-autonomous control is to use a human operator as the feedback controller of the system, it is important not to limit the freedom of the operator to control the system. Thus, it is not meaningful to make predictions that are so far into the future as to allow for enough time for the operator to change their mind and subsequently their planned control input. In practice, therefore, input predictions will be limited by the characteristical time constants in the human motor control system.

Furthermore, by imposing a model on the operator's input, the system is limited to motions that are described with sufficient accuracy by the model. The more specific a model is, the more possible actions would be expected to be excluded, and the more general the model is, the lower the accuracy would be expected for a specific motion. The tradeoff between generality and accuracy will have to be considered thoroughly when designing a system for input estimation, as with any model based estimation.

3.2 Human Motion Models

The human motor system is very complex and not yet completely understood. However, it is well studied, and although not all of the questions as to *why* and *how* humans generate the motions we do may be answered, there is adequate knowledge about what motions we perform.

Trajectory Models for Human Reaching Motion

A subject of study in neurophysiology is how freely moving hand trajectories, such as when reaching towards an object to pick up, are formed in the human motor control system. When moving a hand from one position to another, there are infinitely many possible trajectories to do this. Even when following a specified trajectory, there are infinitely many possible velocity profiles that can be applied to a given trajectory [79]. In the field of neurophysiology, two of the main questions posed regarding the trajectories of reaching motions are:

1. Out of the infinite possibilities, which trajectory and/or velocity profile is chosen?
2. Why is this option chosen?

In the present work, only the first question will be of relevance, but since the two have often been studied simultaneously, many models and attempted descriptions address both questions.

As an early hypothesis, the answer to question 2 was assumed to be found in optimization. Given a multitude of possibilities, it seemed rational that the optimal solution should be chosen. However, the target function of optimization was unknown, and became the first subject of study [36].

One of the first systematic compilations of candidate functions was presented in [107]. A list of plausible functions to minimize was produced and the results attained by applying optimization techniques to these were compared to actual motions. The target functions studied here were:

1. *Total time of the motion, constrained by maximum acceleration.* This generates a trajectory consisting of two second degree polynomials. The first has constant maximum acceleration, the second has constant maximum deceleration.

2. *Peak force applied during the motion.* For negligible friction, this is equivalent to peak acceleration. This generates a trajectory consisting of two second degree polynomials. The first has constant acceleration, the second has constant deceleration, where the acceleration and deceleration are the values needed to reach the target point at the target time.

3. *The absolute value of impulse, as integrated over the duration of the motion.* This is equivalent to the peak velocity attained during the motion. This generates a trajectory that begins with a second degree polynomial with constant maximum acceleration until the peak velocity is reached, then a linear segment with constant velocity, and and finally a second degree polynomial with constant maximum deceleration until the motion stops at the desired point.

4. *Total energy spent during the motion.* This model has the biological rationality that all organisms should tend to conserve energy whenever possible. This generates trajectories consisting of piecewise third degree polynomials.

5. *The value of the jerk (time differential of acceleration) squared, as integrated over the duration of the motion.* This criterion is known as *minimum jerk* (MJ), and was first thoroughly described in [66]. This generates trajectories consisting of a single fifth degree polynomial.

Under the assumption of negligible friction, apart from *total energy*, these can all be expressed in terms of hand position as a function of time. In order to achieve a meaningful definition of *total energy*, one would need some knowledge of the friction characteristics of the arm, as well as the energy conversion characteristics of the muscles.

The study showed that though there were special cases, such as bowing a violin, where minimizing peak velocity gave trajectories that coincided with observations, for unconstrained reaching motions, the best fit was obtained when minimizing either one of total spent energy or the square integral of jerk. These trajectories have a bell shaped velocity profile resembling that observed in actual motion, while the trajectories obtained when minimizing peak velocity or peak acceleration have triangular or trapezoidal velocity profiles. The trajectories obtained when minimizin motion time were very different from actual observations, as most real motions take significantly longer to execute.

The trajectories achieved by minimizing either energy or jerk were very similar, and the energy cost of minimum jerk was less than 2% higher then that of the minimum energy solution. For a free moving body, the MJ solution is the solution that minimizes mechanical strain.

This led to the proposition of a model where the change of joint torque was minimized over the course of trajectory, motivated by claiming that if mechanical strain on the person performing the motion was to be minimized, the hand should not be treated as a freely moving body but as part of the mechanical linkage of the arm. It was also argued that dynamics should be considered, and not only kinematics. Studies showed that for unconstrained reaching motions where body posture underwent negligable change, this model produced the same trajectories as the MJ model. However, when posture was changed significantly during the execution of the motion, or if varying external forces were applied, the minimum torque change model was significantly more accurate [144, 79]. The Jacobian transform from joint-space motion to cartesian motion can be approximated with a linear function locally, meaning that the MJ solution and the minimum torque change solution will be similar. When the motion is large enough that the linearity approximation is no longer valid, the two solutions will diverge.

Yet another physiologically motivated approach is based on information content in the control signal. Observing that the noise in the human motor system is proportional to the signal amplitude, a model where the trajectory of a reaching motion is explained by minimizing the variance of the final position has been suggested. The actual trajectories were generated by applying quadratic programming to find the series of neural input signals that minimized the final variance for a given target point at a given time. While this model may have biological relevance as to explaining why a certain trajectory is chosen, the trajectories it predicts for reaching motions do not differ significantly from those given by MJ or minimum torque change[55].

Choice of Model

In the present work, the usefulness of a model for explaining how the human motor systems performs planning and/or control is of little relevance. Instead, we focus solely on two criteria when choosing an appropriate model for estimating human motion:

1. **Accuracy**. How well does the model fit and/or predict actual observations?
2. **Implementability**. How suitable is the model to implement as a predictor in a real-time teleoperation scenario?

Given the first criterion, we can remove the first simple models that minimize time, peak acceleration or peak velocity, as these do not fit well with observations. If we assume no external forces or large changes of posture, all the remaining models have similar accuracy. For most teleoperation scenarios, this should be a valid assumption since the operator should be assumed to be stationary and using an interface with a relatively small workspace situated comfortably in front of the user.

As for the second criterion, both the MJ model and the minimum energy model can be described by polynomials in cartesian space, but while the MJ model is purely kinematic, the minimum energy model requires modelling the internal dynamics of the arm. The same is also true for the minimum torque change model, and the minimum variance approach.

Thus, by choosing the MJ model, all modelling can be based solely on measurements of the subject's hand position, which can easily be attained with any input device. Also, since this model is polynomial, it is trivial to integrate or differentiate, enabling the use of velocity or acceleration measurements as well as position.

The limitations imposed by the model is that it is only valid in the absence of external forces and when the subject does not change posture significantly. In teleoperation terms, this translates to limiting the application to contactless free motion, with a limited size of the workspace for the input interface.

The Minimum Jerk Model

The *minimum jerk* (MJ) model is well-known and used for explaining the kinematics of visually guided human reaching motions. It was first proposed for single-joint motions in [66], and later extended to include multi-joint planar motion in [41]. It was observed that the trajectory of voluntary arm motions, when described in a cartesian space independent of the subject, follow certain constraints. The trajectories can be predicted by using a model in which the square sum of the third derivative of position, *jerk*, integrated over time is minimized. Thus, given a starting point, an end point and a time to move between the two, the trajectory that minimizes the jerk on this interval is the MJ trajectory. Observations on the motions that humans make when freely catching a thrown ball indicate that they start

by moving towards the expected point of impact with a distinct MJ-type reaching motion, and later add smaller corrective MJ-type motions to accurately catch the ball [56, 88].

All MJ trajectories share the property that the 6th derivative is zero for the duration of the motion, and that they thus can be described as 5th degree polynomials, as in Equation 3.1.

$$x(t) = a_1t^5 + a_2t^4 + a_3t^3 + a_4t^2 + a_5t + a_6 \tag{3.1}$$

If we also add the start and end points of the motion, $x(t_0)$ and $x(t_1)$, and state the position, velocity, and acceleration at these points, we get the following constraints on Equation 3.1.

$$\begin{array}{lll} x(t_0) = x_0, & & x(t_1) = x_1 \\ \dot{x}(t_0) = \dot{x}_0, & & \dot{x}(t_1) = \dot{x}_1 \\ \ddot{x}(t_0) = \ddot{x}_0, & & \ddot{x}(t_1) = \ddot{x}_1 \end{array}$$

The above constraints will give us 6 equations, and we get a well-defined system to find the 6 parameters $a_1 \dots a_6$. Thus, there is only one possible MJ trajectory for a given start and end, and it can be found by solving a simple system of linear equations. For a typical reaching motion, the velocity and acceleration are zero at the start and end points, $\dot{x}_0 = \dot{x}_1 = \ddot{x}_0 = \ddot{x}_1 = 0$. Using this, we can rewrite the equation as a function of a_1 alone:

$$\begin{array}{rl} x(t) = & x_0 + a_1 \left[t^5 - \frac{5}{2}(t_0 + t_1)t^4 + \frac{5}{3}({t_1}^2 + 4t_1t_0 + {t_0}^2)t^3 - \right. \\ & 5(t_1{t_0}^2 - {t_1}^2t_0)t^2 + 5{t_0}^2{t_1}^2t - \\ & \left. \frac{1}{6}{t_0}^5 + \frac{5}{6}{t_0}^4t_1 - \frac{5}{3}{t_0}^3{t_1}^2\right] \end{array} \tag{3.2}$$

Where the remaining constants are related as:

$$\begin{array}{lll} a_2 & = & -\frac{5}{2}(t_0 + t_1)a_1 \\ a_3 & = & \frac{5}{3}(t_0^2 + t_1^2 + 4t_0t_1)a_1 \\ a_4 & = & -5(t_0^2t_1 + t_0t_1^2)a_1 \\ a_5 & = & -5a_1t_0^4 - 4a_2t_0^3 - 3a_3t_0^2 - 2a_4t_0 \\ a_6 & = & x_0 - (a_1t_0^5 + a_2t_0^4 + a_3t_0^3 + a_4t_0^2 + a_5t_0) \end{array}$$

Without loss of generality, we can choose the coordinate system so that the start position $x_0 = 0$, and that the motion starts at $t_0 = 0$ and the equation can then be rewritten as:

$$\mathrm{x}(t) = a_1(t^5 - \tfrac{5}{2}t_1t^4 + \tfrac{5}{3}{t_1}^2t^3) \tag{3.3}$$

If we solve for a_1 at $t = t_1$ we get:

$$a_1 = \tfrac{6x_1}{{t_1}^5} \tag{3.4}$$

So the entire trajectory is defined completely by the distance (x_1) and duration (t_1). An illustration of a typical MJ trajectory and its first five derivatives (velocity, acceleration, jerk, snap, and crackle) are shown in Figure 3.2. In this case, the motion has duration 0.5 s and the distance moved is 0.3 m.

For 3-dimensional motion, each dimension of the motion is described by a polynomial as in Equation 3.1, where the coefficients for the different dimensions are independent from one another, but the t_0 and t_1 are the same [79].

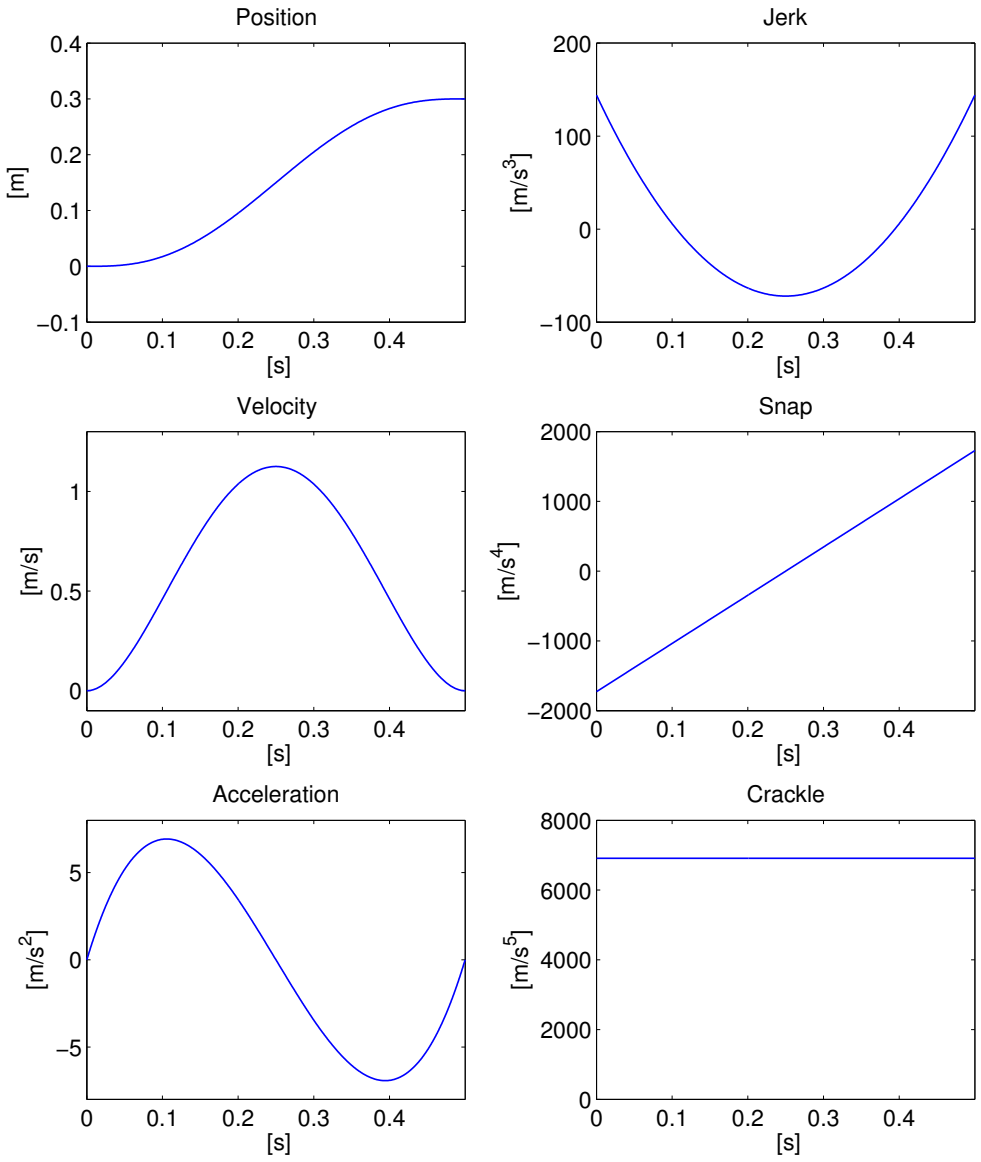


Figure 3.2: An illustration of a typical MJ trajectory and its five first derivatives. The names "snap" and "crackle" are sometimes playfully used for the fourth and fifth derivatives of position, and are used here as no other names have formal status.

Superposition Priciple

The trajectories described by a simple MJ model are limited to one single MJ motion. It has been shown that each MJ motion is executed in a feed-forward manner without feedback [17]. If a more complex motion is performed, or if the target of the motion is changed in mid-motion, the trajectory can be described by generating a new MJ submotion before the old motion is ended and superpositioning this onto the old [103]. If the added MJ trajectory has an initial position, velocity, and acceleration of zero, this will still result in a continuous, smooth (first two differentials continuous) motion where the 6th derivative is zero, so the jerk is still minimized. For these types of compound motions, a common trait is that the tangential velocity tends to be lower as the radius of curvature decreases. Thus, if the direction of a new motion differs significantly from that of the previous motion, the new motion will not be added until the previous motion is close to finishing. Compound MJ motions have been described thoroughly in [67, 40, 146, 47]. In the motor control system of humans, a new submovement can be generated as often as once every 100 ms [100]. This observation, in combination with the feed-forward nature of the individual submotions, makes it reasonable to assume that human motions could be possible to predict accurately for att least 100 ms.

As an illustration, we fit MJ trajectories to actual human reaching motions. The human motions were recorded in an experiment with teleoperated ballcatching in a virtual environment, as described in Section 6.1. In the first example, the subject was successful in determining where the ball would come, and caught the ball with a single reaching motion. Figure 3.3 shows the largest component (y) of this reaching motion, with a single MJ trajectory fitted.

In another try, the subject was not initially successful in determining the trajectory of the ball, but had to make subsequent corrections of hand position in order to successfully catch it. Figure 3.4 shows the largest component (y) of this reaching motion, with a series of MJ trajectories fit in such a way that the superposition of the submotions fits the measured position.

3.3 Related Work

There have been a few applications using minimum jerk models to estimate the input from a human operator, and minimum jerk models have been used in a multitude of other applications to robotics.

Recently, Weber et. al. have presented work where MJ models are applied to user input in a teleoperated system. Their approach limits robot motion to avoid collision, by substituting the user's input signal with the MJ trajectory that most closely resembles the original input, while not colliding with a wall. Their approach has only been applied to 1 DoF motion, but shows that it is possible to use human models to limit robot motions to a safe workspace without removing the sense of presence of the operator, as the limited motion is not perceived as being in conflict with user's intended motion [148].

Jarrassé et. al. have studied the use of human motion prediction to enhance the perceived transparency of a teleoperation system. However, they do not explicitely treat the prediction process itself, but assume that prediction can be done, and substitute prerecorded motion data for predictions for a task where the exact motion profile is known beforehand [75].

Apart from these, one of the most widely studied uses of human motion models in robot control is to generate as humanlike motions as possible for robots with higher degrees of autonomy. There exist three main motivations for these applications.

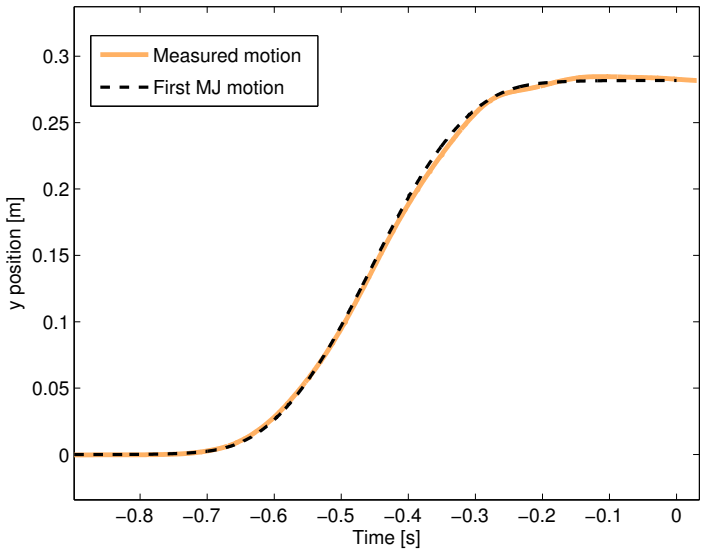


Figure 3.3: One of the components (y) of the measured hand trajectory with MJ trajectory fitted. In this case the hand trajectory contains only one major MJ component.

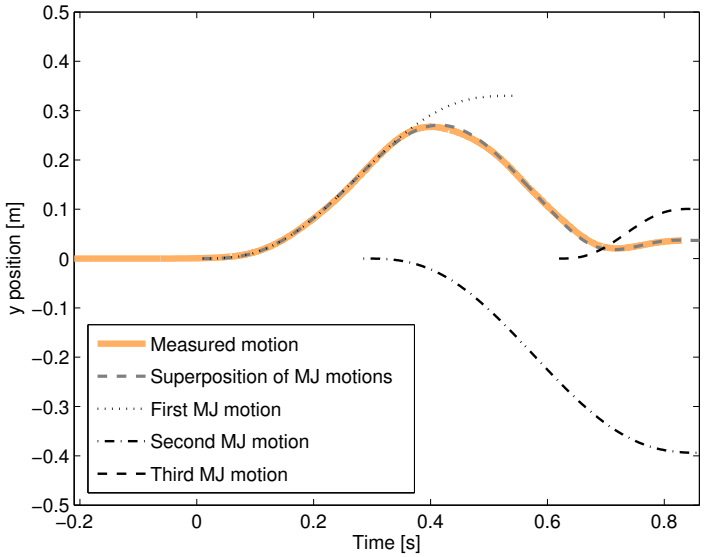


Figure 3.4: One of the components (y) of the measured hand trajectory with MJ trajectory fitted. In this case the hand trajectory contains two major MJ components and one minor.

The first motivation is that some robots are aimed for applications that require them to have as humanlike behavior as possible, such as humanlike androids or robotic prosthesis replacing human limbs. MJ trajectories have been used to generate humanlike motion for prosthetic fingers [122]. They have also been used as a smoothing function for a path planner for a humanoid robot, with the auxillary goal of producing motion that is as humanlike as possible [89]. A slightly different approach is to mimic human gaze shifts during manipulation, by using humanlike anticipatory motions to align a camera for teleoperation. This has shown to improve both objective performance and perceived immersion [119].

The second motivation is for robots working in close proximity to humans, where having the same motion profiles as a human makes interaction easier. This would for example be industrial robots that handle an object together with a human, for instance making the load lighter while the human controls the trajectory. Corteville et. al. have proposed a system that uses an online estimator based on minimum jerk models for *admittance control*. This system uses the estimate of the human operator's input to generate forces that moves the robot along the same trajectory as the human, thereby aiding the humans's motion. So far, experiments are limited to 1 DoF motion, but the approach is thought to be applicable to more general motions. Proposed applications include lifting aids for moving heavy loads. A successful application should minimize the force that the operator has to exert to move the load [32].

A similar motivation has led to minimum jerk trajectories being used to generate smooth interaction for robot-assisted therapy for recovering stroke victims, where the robot enforces MJ motions while helping the patient to move his/her arm, mimicking the actions of a human physical therapist [86].

The third motivation is the common motivation of biomimetic design, that if it is good enough for a human, then it must also be good enough for a robot, or rephrased: "if evolution determined that MJ and MJ-like motions were the most efficient and wear-minimizing, who are we to argue?" A control law that generates minimum jerk trajectories to optimize precision is presented in [91]. A system utilizing a neural network and minimum torque change to learn to control an articulated arm without explicit knowledge of kinematics and dynamics was proposed in [78]. Observations that the superposition strategy works well for humans when replanning motion led to applying the same to robot on-line path-planning [47]. A biologically inspired MJ-based control law for robot manipulators with the aim to reduce wear was presented in [113].

Using much simpler models of human motion, that do not use trajectory models, teleoperation schemes have been proposed that include simplified models of the passive impedance of the operator. These typically contain the human as a spring-damper object, and are mainly concerned with the reactive aspect of the operator's interaction with a force-feedback device. These do not try to model the proactive control inputs from the operator — these are either treated as noise or as the given control signal u [53].

Also related is work that models the human behavior at a higher level. This is sometimes called *intent detection*, and typically models the input at a task level rather than precise motions. An example is using statistical models to infer which of a number of possible paths a user intends for a walker in the presence of obstacles [70]. There have been attempts to classify user input and map reaching motion to to the most probable target for space telerobotics [141].

There exists an implementation for pick-and-place tasks for a humaoid torso, where the system estimates the intended target by measuring the jointspace distance between user input trajectory and an autonomously generated trajectory for each possible target, and lets the user switch from direct control to autonomous mode when he/she accepts the proposed target of the system [37]. Work has also been done on using Hidden Markov

Models to identify the operator's current intention, with the aim of switching control modes between "fast" or "accurate", depending on type of current operation [71].

Chapter 4

Experiment Design

We want to try the ideas presented in the previous chapter with a series of experiments. We thus need to find relevant experimental tasks and setups. This chapter arguments for what type of experiments should be done and why.

4.1 Tasks

The validity of the minimum jerk (MJ) model described in Section 3.2 for estimating operator input can be evaluated by applying it to different teleoperation scenarios. In order to have a thorough testing, it is reasonable to both apply the methods to scenarios that should be a perfect match for the MJ model's strengths and weaknesses and to more general scenarios where the MJ model might not be obviously appropriate.

A teleoperation task where the MJ model would be expected to be a good description of user input should fulfill the following criteria:

1. The task should be visually guided, since most experimental support for the MJ model is based on visually guided hand motions.

2. There should not be any significant contact forces acting in the direction of motion, as these would alter the velocity profile.

3. The task should be accomplishable in a single reaching motion, so that it can be accurately described by a single MJ submotion. This will strip the experimental treatment of the problem of handling superpositioning of several MJ submotions.

4. The task should require fast reactions, forcing the operator to perform MJ-type feedforward motion. Also, the potential benefit of a successful application of input estimation is larger for tasks that require fast execution.

The first two criteria are necessary for the MJ model assumptions to hold, but the last two criteria could be changed in order to make a more challenging test:

3. The task should require continuous motion, so that a large number of superimposed submotions are required to describe the trajectory.

4. The task should allow for some forward planning for the operator, so that the operator is free to choose any possible trajectory.

Furthermore, in order for the experiment to be conclusive, we should use a task that requires fairly high precision to be performed successfully. Thus, a model-based estimator that does not fit the user input well will be likely to cause a failure to complete the task, and can thereby be more easily identified.

Based on these criteria, we suggest three basic tasks to test in experiments.

Reaching to Touch a Target

A first task, that is prototypical for the MJ model, is to reach towards a stationary target. This means that the subject has clear view of the target and can perform a simple reaching motion to touch the target. This type of task is mostly a proof-of-concept type task. If the MJ model is at all applicable, it should work for this task, as explained in Section 3.2. This task can also be used to test and design the predictor implementation.

Ballcatching

To further test the single reaching type motion, we also try a task that is inherently difficult for the user to perform successfully without any aids. The purpose of this is to find if a significant improvement can be made with the use of MJ models. We propose ball-catching as the second task. Robot manipulators have been made to autonomously catch thrown balls since the early 1980's [4, 69, 44], so the task should be physically possible. However, given the necessary reaction time and precision, it should be difficult for a human operator to achieve.

There is a multitude of literature on human ballcatching, such as in sports and games. The possible strategies for ball-catching can be divided into two categories, *predictive* strategies relying on ballistic models, or *prospective* strategies utilizing feedback [35]. Predictive ball-catching relies on a ballistic model of ball motion and observed initial conditions to make a prediction of the most probable trajectory. A predictive strategy is necessary when the motion commands have to be started well before the point of impact, i.e. when the time of flight of the ball is short compared to the reaction and/or motion time of the catcher. This is typical for batting in baseball or cricket [35, 92]. Prospective strategies utilize continuous feedback from the task performed. There is no explicit need for an exact model of the target's motion, but instead, corrections are made to minimize the distance or relative velocity between target and catcher. This type of strategy is viable when there is enough time to perform several corrections before the point of impact, and is useful when the trajectory is difficult to predict from initial conditions. This is typical for outfield players in baseball [35].

Catching of balls that have been thrown for shorter distances — where the viewpoint location is clearly separate from the position of the hand — has been studied in terms of the internal models that humans might apply to the catching task. Specifically, it has been shown that a priori knowledge about gravity direction and magnitude is used when catching falling balls [99]. When catching a ball that is thrown across a room at a moderate velocity of approximately 5 to 6 m/s, the time for the entire throw is typically second or less[1]. In one study, it was found that the average hand motion time for a human subject attempting to catch such a ball was 0.52 s, and that the average hand trajectory length was approximately 0.35 m [88]. In these cases, there does not seem to be enough time for a conscious continuous feed-back loop to correct the hand position, but rather an initial estimate is made of the

[1]The time of flight is limited by ceiling height. The higher the apex of the trajectory is above the start and end points, the longer the ball will be in flight. For a throw where the apex is 1.3 meters higher than the start and end points, as could be the typical limit for an indoor throw, the time of flight is 1.03 s.

ball trajectory, and the hand moved roughly toward a point of intercept. If there is still time to react, this position can be corrected to produce an accurate catch, possibly applying multiple corrections. Evidence towards this is that one can distinguish a distict trajectory towards an initial catch position estimate, and additional shorter distinct trajectories that correct this [56].

Linetracing

Finally, in order to have a task that is as general as possible, we wish to let the operator move the arm for extended continuous motions. Since we want the motion to be visually guided, and we also want to have a ground truth by which to compare the performance, we let the operator use the robot arm to follow a preset pattern on a piece of paper. This should generalize to many other types of complex visually guided tasks.

4.2 Interface Types

There are several types of interfaces that are interesting to study. First, in order to assure that the motion models are valid, it is desirable to use an interface where the human operator has a high level of immersion, and is allowed to make as natural movements as possible, i.e. there are no significant interaction forces. The results gained with such an interface can then be compared to the results with a more standardized teleoperation interface in order to evaluate what parts of potential performance problems can be attributed to the interface.

It is also interesting to use a very simple and/or cheap interface where the quality of the input signal may not be very high, in order to test the possibilities to overcome poor measurements using the modelbased estimation approach.

Given that one of the most significant problems facing any teleoperation scheme is transmission time delay, we should also have a setup that can address this. However, one of the main reasons why time delays are a substantial problem is that they are present in virtually any advanced system. Thus, it should be trivial to introduce time delays into any setup regardless of the interface type, and this will not be a major issue in the choice of interface.

With these motivations, we shall study the following types of interfaces:

Virtual Reality

The human motion models we intend to use originate from studies of unconstrained human subjects reaching freely for visible targets. To recreate this as accurately as possible using available technology, we propose to use virtual reality (VR). We can let the the subject move freely and use non-contact measurements for handtracking to generate input signals, and give visual feedback via a head-mounted display (HMD). This allows for free 3D motion.

VR with a HMD has been used to enhance teleoperation and has been shown to give unrivaled levels of immersion [24]. VR as a technology is mature enough that behavioral scientists consider it usable to perform experiments and consider the results as applicable to the real world [142].

Some concerns have been raised regarding the safety of VR systems. VR with HMDs has been connected to some motoric (balance and other) symptoms after usage, but sufferers are few and recover quickly and completely [30].

Screen and Joystick

We also want to try a setup that is simple to implement and easy to replicate by others, so that our results will be valid for standard type setups. One of the most common types of interfaces for teleoperation is a joystick for input and a computer screen for visual feedback. The joystick can have 1 or 2 DoFs in simpler setups, like reported for space teleoperation and urban rescue robots [72, 26, 42], or 3 or more DoFs for more advanced motions for advanced experimental use [28, 85]. For better immersion, the computer monitor can show 3D images by using shutter glass technology, as demonstrated in advanced teleoperation experiments in orbital satellites [63].

Game Controller

Finally, we also want to examine the potential use of model based input estimation on less-than-perfect input signals. Recently, mass market video game controllers have begun to attract attention as teleoperation interfaces. Specifically, the remote control device for the Nintendo Wii video game, the accelerometer-equipped "wiimote" interface is reported as being intuitive and easy to use [52, 93, 46, 145, 98, 128]. Even though it is far from as accurate as a traditional teleoperation control interface, the cognitive load is lower, allowing the user to concentrate on other things, or performing several tasks simultaneously. Others report that wiimote control is less precise and lowers task performance as compared to more traditional input devices [138]. Recently, an industrial robot arm controlled with a wiimote has attracted media attention [31], but so far this has only been using task level control where the task is not performed until the user input motion has ended. To the knowledge of the author, there are no reports of direct position tracking using the wiimote accelerometers, so an implementation that enables position tracking is also an interesting demonstrator application in its own right.

Chapter 5

Design of Experimental System

This chapter describes the systems used for all tests and experimemts presented in this thesis. Custom built parts of the systems are described in more detail than those that are readily commercially available.

5.1 Robot Manipulator

Given the physical decoupling of master and slave that is inherent in most teleoperation systems, many scenarios could probably be tested with a completely simulated slave environment, removing the need of an expensive and cumbersome physical robot. However, if we were to create a completely virtual system, the suspicion that the simulation is less complex than a real setup is difficult to shake. Therefore, in order to make sure that no unintentional simplifications are made, and that in extension the results are applicable for real systems, it is preferable to use a real physical robot.

Since the experiments we want to perform concern different types of reaching, touching, and catching motions, it is natural tu use a stationary robot manipulator, rather than a mobile robot. The choice of robot is non-trivial given the amount of money and time involved in setting up a new system. Therefore, it is important to start by making rigorous specifications. In our case, we want the robot to be fast enough to match the operator's motions, so that we can minimize the problems associated with motion discrepancies between master and slave, as described in Section 2.1. We also want to implement our own controller so that we can perform controlled experiments. We may need to simulate the robot to predict observations, so we want to have accurate models of the robot dynamics, which requires access to descriptions of both hardware and software. Unfortunately, commercial systems rarely provide such well-documented low-level access.

A large number of robot manipulators have been designed over the last half century, and several of these have become standard platforms for R&D efforts. Historically, the most widely used for academic research is without a doubt the Unimate PUMA 5xx series, which is readily available, but lacking in dynamic capacity. As actuation systems have become more powerful and miniaturized it has become possible to build fast robot systems to perform highly dynamic tasks. Early examples of highly dynamic robot control include the ping-pong playing robot at Bell Labs [3], and the juggling robot developed by Koditschek et al [23, 116]. Another example of dynamic systems are walking robots [115].

Several commercially available candidates were examined - see Table 5.1. Of these, the only candidate that fulfills both performance and accessibility requirements is the KUKA LWR [62], which was not commercially available when this research was initiated (it was

Table 5.1: Comparison to some alternative manipulators. Data as given in manufacturers' documentation.

name:	price(€)[a]:	dof:	reach:	weight:	power[b]:	payload:	API:	velocity:
Puma 560	—[c]	6	0.86m	63 kg	1.5 kW	2.5kg	RT joint	0.5 m/s
Neuronics Katana	20 000	5	0.60m	4.3 kg	96 W	0.4 kg	RT traj/joint	90 deg/s
KUKA KR5 850	22 000	6	0.85m	29 kg	2.3 kW	5 kg	80 Hz pos/vel	250 deg/s
Schunk LWA3	45 000	7	—[d]	10 kg[d]	0.48 kW	5 kg	traj/current	70 deg/s
Custom Robot[e]	50 000	6	0.91m	23 kg	5.5 kW	—[f]	600 Hz pos/vel	7m/s
Barret WAM	70 000	7	1 m	27 kg	250 W	3 kg	500 Hz traj/force	1 m/s
KUKA LWR	120 000	6	0.94m	14 kg	720 W	14 kg	1 kHz force/traj	120 deg/s

(a) Actual prices paid or as quoted by supplier.

(b) Rated peak power consumption.

(c) Used, price varies with condition.

(d) Available in different configurations.

(e) The manipulator described in the present text.

(f) Not tested for durability.

announced available on the Euron Mailing list on Aug 25, 2009), and had an expected price that at the time was prohibitive. A light-weight industrial robot like the KUKA KR5 may have a more attractive performance/price ratio, but the proprietary interface limits control to high-level position or velocity control at 80 Hz with no low-level interface, meaning that it is not suitable for our research applications

This left the alternative of constructing a custom built robot, like many of the examples used for dynamic manipulation experiments mentioned above. This raises the concern that in the field of robotics, too much research is performed on a basis that cannot be replicated, reproduced or reused. Recently there have been several attempts to utilize standard platforms for research, in order to assure repeatability of experiments, as exemplified by the LAGR program organized by DARPA [74]. The RobotCub project has also made a few robots available to the research community [121]. Given this background, it is an interesting inquiry in its own right to see if an inexpensive high performance manipulator can be custom made from standard parts, so that it is easy for anyone to replicate.

Basic Requirements

The most demanding type of experiment that we want to perform involve catching a ball thrown across a room. We anticipate a normal, slow, underhand throw from a distance of approximately 4 m. In an indoor environment, a ball can be thrown with reasonable accuracy along a parabolic path with an apex of 2.3 m, with both the thrower and the catcher situated at a height of approximately 1 m (see Figure 5.1). Simple studies of human performance indicates that the system must be able to accommodate variations in thrower accuracy corresponding to catching the ball within a 60×60 cm window. From these basic requirements it is possible to compute flight time and velocities for the scenario, as summarized below:

- Throwing distance will be approximately 4 m.
- Flight time will be up to 1 s, the typical time is expected to be 0.8 s.
- The ball will travel with an approximate velocity of 6 m/s at time of arrival.

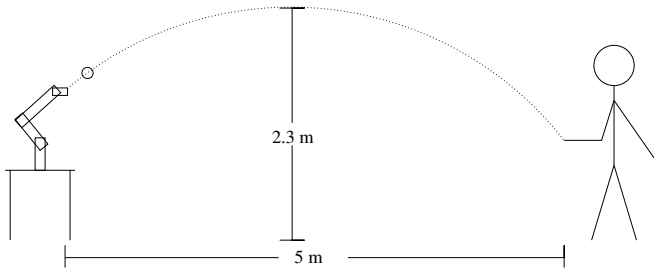


Figure 5.1: Schematic of ballcatching experiment.

- The ball should be caught if it arrives within a 0.6 m × 0.6 m window.

One desired feature is the use of standard video cameras for ball trajectory estimation. With normal 50 Hz cameras, the frame time is approximately 20 ms, and a similar time window is expected to be needed for segmenting the ball in the image and estimating the position. In addition, at least three frames are required for trajectory estimation, but limited accuracy of the cameras will mean that more, probably as many as 10 images might be necessary (c.f. [44]). Thus, the time delay from the initiation of a throw to the initial trajectory estimate might be 200 ms. The setup is intended to be used for teleoperated catching, were a human operator steers the robot towards the predicted trajectory, so we have to allow for the operator's reaction time as well. This might be around 100 ms, so a window of 300 ms is reserved for initial reaction to a throw, leaving 500 ms in which the arm has to move into position. In the worst-case scenario, the arm has to move from one opposing corner of the operational window to another, a distance of almost 0.9 m. Since the experiments will not be concerned with grasping, a simple passive end effector like a small bucket can be employed, and the positioning error has to be smaller than the radius of the bucket, preferably less than 1 cm. These requirements can be summarized as:

- End effector has to be able to move 0.9 m in 0.5 s, (partially) against gravity, from stand-still to stand-still.

- The precision of positioning the end effector should be within 1 cm

Given constant and equal acceleration and deceleration, a distance of 0.9 m can be traveled in 0.5 s if the acceleration is at least 14.4 m/s^2, and the maximum velocity is at least 3.6 m/s. This also has to be achieved when working against gravity. These are the minimum dynamic requirements — the actual implementation should have some margin to allow for uncertainties. To enable flexibility in the design of future experiments, it is desirable to allow for different types of sensors to be mounted in the end effector reference frame, so this should be freely orientable in a dexterous manner. The end effector therefore has to have 6 degrees of freedom, and should be freely orientable within the entire operation window.

The system thus requires significant dynamics and the control has to be performed in real-time. This implies that it is desirable to have closed form solutions for kinematics to avoid computationally expensive numerical calculations, which in turn imposes constraints on the design of the overall kinematic structure. Without a closed form kinematic/dynamic solution it would be much more challenging to guarantee the real-time performance.

A highly dynamic robot arm will pose a potential hazard to both its operator and itself unless sufficient precautions are taken. Therefore, the control of the arm has to be sufficiently exact so that safe paths can be accurately followed, and precautions against malfunctions have to be taken. The former requires control loops running at a high frequency/low latency, the latter that software and hardware malfunctions are kept at a minimum, and that the negative effects of malfunctions should be minimized. Thus, the software environment has to be a stable real-time system, while the hardware contains fail-safe fallback for dealing with software failure.

These further requirements imposed on the design are summarized below:

- Closed form analytical kinematics and dynamics are necessary for speed of calculations.
- At least 6 degrees of freedom.
- Acceleration of at least 14.4 m/s^2 for end effector.
- Velocity of end effector of at least 3.6 m/s.
- Safety for operator and machinery requires a stable real-time system, as well as fault-tolerant hardware.

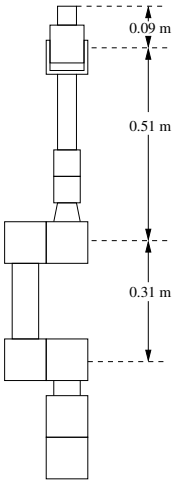
Designed Solution

Having decided to construct our own 6 DoF arm, we examine if this can be done using *PowerCube* modules from Amtec[1]. These modules are available off the shelf and allow rapid prototyping. The range of modules clearly include some that have specifications which are adequate for the target application (See Section 5.1). These modules also have a built-in controller that can be used for embedded safety functions.

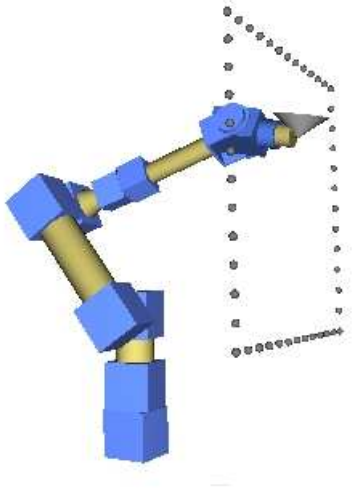
The actual performance depends on the configuration that the modules are assembled in, so a few different configurations were examined more closely in computer simulation, where a 10 % uncertainty was added to the maker specifications. The configuration that showed the most promising results is one that is kinematically very similar to a Puma560 arm (and to many other commercially available robots). This is not only a configuration that allows for very good dynamic performance (see Section 5.1), but as it has been widely used and studied, several implementation issues are already solved, thus making the design process considerably faster. For example, the closed form solution for inverse kinematics and dynamics are well-known.

The choice of gearings and link lengths induce a trade-off between acceleration and end effector velocities. The balancing of this trade-off has been made to minimize the time needed to move the end effector from one stationary position to another within the operation window. Since there is a limited, discrete amount of possible combinations of actuators, it was possible to find the optimum through an exhaustive search. The resulting configuration that performed the best in simulation is specified in Table 5.3. The design and dimensions can be seen in Figure 5.2. The design allows for a workspace that is more than large enough to accommodate for the specified 60 cm × 60 cm window for ballcatching, though the manipulator's dynamic performance deteriorates somewhat at the edges of the workspace. A cross-section of the workspace can be seen in Fig. 5.2(c). The arm has rotational symmetry as viewed from above, but is for safety reasons limited to a half-circle to avoid collisions with any objects behind it.

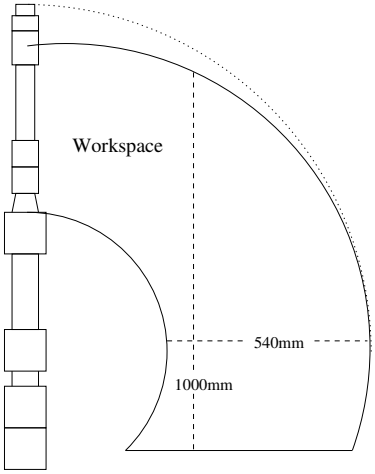
[1]Now Schunk, Ltd.



(a) Dimensions



(b) 3D rendering of arm and operational window



(c) Workspace with tool oriented towards user.

Figure 5.2: The first design of the manipulator, using Amtec PowerCubes.

The PowerCube modules support several different communication protocols, but for robustness and responsiveness, the option of a 1 Mbit/s CAN bus was deemed optimal. The hardware design can be implemented in several different ways. In principle all modules could be on a single CAN bus or each module could have a bus of its own. The end-effector joint is a combined pan-tilt which requires use of a single bus to control both degrees of freedom. This means that the control computer could be equipped with either 1, 2, or 3 CAN controllers for symmetric loads, or 4 or 5 controllers for assymetric loads, where the inner joints that control positioning are run at a higher frequency than the outermost controlling orientation. Simulations where the inner joints were controlled at 500 Hz and

the outer joints at 200 Hz show that this is a viable option. In simulation, the inner joints can be stably controlled at full power output using frequencies from approximately 400 Hz and upwards.

The first choice for the operating system for the computer performing the direct low-level control of the robot was RTAI, a real time GNU/Linux system that has showed good performance in previous studies [1], but some testing led to the choice of a standard GNU/Linux kernel patched with high resolution timers[2], as this not only showed at least equal realtime performance but allows for easier implementations in user space. The control computer will also perform the trajectory generation and be responsible for dynamic and kinematic calculations.

Specifications

The basic design specification is outlined below:

- 6 DoF arm made with Amtec PowerCubes
- Kinematic configuration of *Puma560* type
- GNU/Linux, preferably with high resolution timers, for control computer
- Communication over several parallel CAN connections.

Simulated Performance

The performance of the proposed arm was first calculated using the specifications from Amtec and numerical simulations. The results for the travel times depend on the type of controller used, and in this particular case, the controller included a dynamic model for feed-forward control, and the torques in each individual joint were set in order to achieve a target velocity as quick as possible. The target velocity was chosen as the minimum of the actuators maximum velocity and the highest velocity from which stopping at the desired position was achievable. This latter factor was calculated using the maximum torque of the actuators and the inertial load of the current configuration, a figure that was multiplied with a factor slightly less than 1 to achieve a margin. Using this simple controller, the simulated arm had more than adequate performance, as is summarized in Table 5.2. Note that the first acceleration figure given is the maximum achievable for small movements, and the second figure for larger, cross-workspace movements.

Hardware Implementation

The arm proposed and specified in the earlier sections was constructed and mounted on a sturdy industrial work table (see photo in Figure 5.3). The lower three actuators have a maximum output of almost 1.5 kW each, harmonic drive gearboxes and incorporated brakes to lessen motor strain when not moving. The fourth actuator is similar, but considerably smaller as it carries a lighter inertial load. The maximum output is 0.36 kW. The last two joints are contained in a combined pan/tilt unit. This is less powerful, but has lower weight per joint than other solutions. This also incorporates the same gearbox and brakes as the other modules. Specifications can be found in Tables 5.3, 5.4 and 5.5

The PowerCube modules have a simple onboard controller that also implements basic security features. They will not allow motion beyond certain angle limits (that can be set

[2] `http://www.tglx.de/hrtimers.html`

Table 5.2: Simulated performance of robot arm. All values are given as their lower limit within the operational window, and are thus equal to or better than this in the entire window. The first acceleration given is the maximum possible for small movements, the second (in braces) is the maximum acceleration achievable for large movements.

Endpoint acceleration	$> 100\ m/s^2 \quad (> 30 m/s^2)$
Endpoint velocity	$> 5\ m/s$
Traveltime across window vertically, from standstill to standstill	$< 0.36\ s$
Traveltime across window diagonal, from standstill to standstill	$< 0.37\ s$
Traveltime from window center to upper corner, from standstill to standstill	$< 0.22\ s$
Repeatability of position	$\pm 1\ mm$

Figure 5.3: The custom built manipulator.

Table 5.3: Specifications for the parts used in the manipulator.

Part	Product name	mass	Comment
1st joint	PowerCube PR110	5.6 kg	51:1 reduction gear
1st link	PAM104	0.2 kg	55mm cylindrical rigid link
2nd joint	PowerCube PR110	5.6 kg	101:1 reduction gear
2nd link	PAM108	0.8 kg	200mm cylindrical rigid link
3rd joint	PowerCube PR110	5.6 kg	51:1 reduction gear
3rd link	PAM119	0.2 kg	45mm conical rigid link
4th joint	PowerCube PR070	1.7 kg	51:1 reduction gear
4th link	PAM106	0.6 kg	200mm cylindrical rigid link
5th,6th joint	PowerCube PW070	1.8 kg	2DoF wrist joint

Table 5.4: Manufacturer's specifications for the joint actuators.

Joint no.	max torque	max angular velocity	repeatability
1	134 Nm	8.2 rad/s (470^o/s)	±0.00035 rad
2	267 Nm	4.1 rad/s (238^o/s)	±0.00035 rad
3	134 Nm	8.2 rad/s (470^o/s)	±0.00035 rad
4	23 Nm	8.2 rad/s (470^o/s)	±0.00035 rad
5	35 Nm	4.3 rad/s (248^o/s)	±0.00035 rad
6	8 Nm	6.2 rad/s (356^o/s)	±0.00035 rad

Table 5.5: The Denavit-Hartenberg parameters for the arm, using J.J. Craig's notation [33].

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0°	0 m	0 m	θ_1
2	−90°	0 m	0 m	θ_2
3	0°	0.31 m	0 m	θ_3
4	−90°	0 m	0.51 m	θ_4
5	−90°	0 m	0 m	θ_5
6	90°	0 m	0 m	θ_6

by the user), and will perform an emergency stop if these limits are exceeded, or if no watchdog signal has been transmitted over the CAN bus for 50 ms.

In order to avoid collisions, both with itself and environment, the angles of the different joints have been limited to the values shown in Table 5.6. There are two sets of limits, each set prohibiting collisions in itself but with a limited workspace. The system will switch limit sets when moving out of range of one set and into range of another, with an intermediate limit set that consists of the tighter limits of the two sets. This means that even if communication would break down in the middle of a limit switch, the individual modules will always be limited to safe intervals, while at the same time allowing for use of a large part of the robot's potential workspace.

To test these safety measures, two experiments were conducted. In the first experiment, the communication link was severed between the computer and the robot. This results in a termination of the watchdog update, and the modules finish their last command and engage the brakes. In the second security experiment, illegal position commands were intently issued by the control program. The modules' onboard controller correctly identified these as violating joint limits. The arm moved into the legal position that was closest to the commanded position and stopped. This should account for safe handling of an unexpected breakdown of control algorithms, the control computer, or the CAN communication link.

Table 5.6: Limits on joint angles.

Joint no	set 1	set 2
1	-90^o – $+90^o$	-90^o – $+90^o$
2	-100^o – -40^o	-130^o – -70^o
3	-60^o – 50^o	-40^o – 90^o
4	-160^o – $+160^o$	-160^o – $+160^o$
5	-120^o – $+120^o$	-120^o – $+120^o$
6	-180^o – $+180^o$	-180^o – $+180^o$

The communication interface was designed to be implemented over 4 separate CAN buses, one each for the 3 inner (position controlling) joints, and one common bus for the 3 outer (orientation controlling) joints. Two 2-channel PCI CAN controllers from Kvaser were chosen, as these had open source device drivers for Linux that where deemed plausible to port to real-time usage.

Software Implementation

A Linux 2.6.19 Kernel was patched with high resolution timers for low latency realtime performance. A customized communications API was implemented to guarantee low-latency communication with the PowerCube modules, as well as customized libraries for fast vector manipulations optimized for calculating arm dynamics. The control loop is run in soft real-time, using the round-robin scheduler `SCHED_RR` and maximum priority. All services not necessary for robot control are turned off. Experiments running the robot for several hundred hours have shown that this gives a worst case latency of less than 100 μs, which is sufficient. The average jitter for the main loop of the control algorithm is 6 μs, which is significantly less than the modules' latency of up to 600 μs. For this application, this soft real-time performance is comparable to that of hard real-time systems like RTAI, but with the advantage of simple straight-forward userspace implementation.

Inverse kinematics and dynamics are calculated using a C implementation of the analytical solution for a Puma arm in [33], and the forward dynamics are calculated using the second algorithm in [147]. As a result, inverse kinematics can be calculated in 1.7μs, and dynamics in 41μs, so that all calculations needed in the control loop take less than 50μs. This means that virtually all latency in the control loop originates from the CAN bus communication path and the PowerCube modules response times.

Combined position and velocity control has been implemented on the system using a combined feed-forward computed torque control (CTC) scheme and a feed-back PI controller. When a new setpoint enters the controller, a velocity ramp trajectory is calculated in joint space. This trajectory is limited by a preset top velocity (presently 4 rad/s) and a maximum acceleration, but otherwise is the shortest path towards reaching the desired

position θ_d and velocity $\dot{\theta}_d$ from the actual position θ_a, without exceeding the maximum allowable acceleration a_{max}, see Equation 5.1. This ramp works under the assumption that the desired position is frequently updated to new positions in accordance with the desired velocity.

$$\dot{\theta}_{ramp} \;=\; \sqrt{|\theta_d - \theta_a| \cdot a_{max}} \;\cdot\; sign(\theta_d - \theta_a) + \dot{\theta}_d \tag{5.1}$$

The maximum acceleration a_{max} is limited by a preset limit value[3] and the maximum achievable acceleration, computed by calculating the resulting acceleration with maximum torque and taking away a small safety margin. The desired acceleration fed to the the CTC controller is then the acceleration necessary to achieve the target velocity $\dot{\theta}_{ramp}$ as soon as possible, without violating the limits on acceleration or jerk. For use in the experiments the acceleration is limited to 16 rad/s^2, and jerk to 400 rad/s^3.

The ramp trajectory is recalculated in each iteration of the control loop. The current position and velocity, and the acceleration prescribed by the ramp, are fed to the inverse dynamics function that determines the necessary torques to follow the trajectory. These torques are converted to currents assuming a linear conversion factor and sent to the actuator modules.

A corrective term consisting of a PI controller monitors the difference between desired velocity and actual velocity, and corrects the controller current accordingly. This term is necessary as the feedforward CTC controller does not contain an accurate enough model of friction, the movements of the power cords or the nonlinearities in the current/torque relationship. For a schematic of the control scheme, see Fig 5.4.

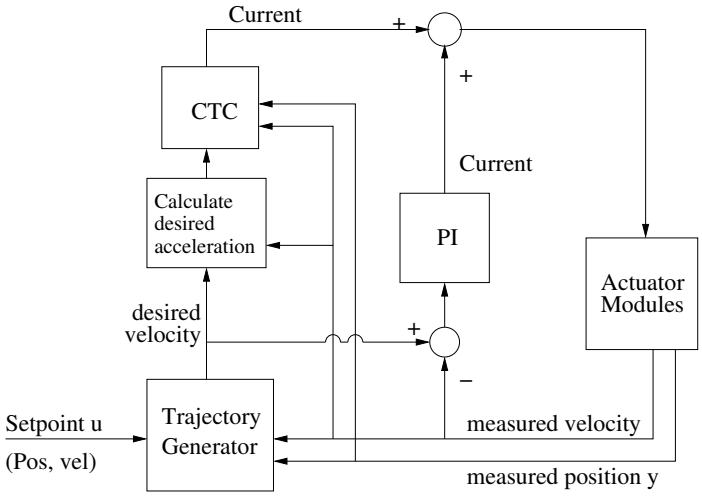


Figure 5.4: Schematic of controller

[3]The limits on velocity, acceleration, and jerk are chosen to limit the mechanical stress on the system, while still being able to reach a given point in the workspace in less than 0.5 s.

Table 5.7: Prices (in Euro) for the robot arm setup

Part(s)	Price (€)
Actuators	38,000
Rigid Links	3,400
CAN System	1,600
Mountings	500
Power Supply	4,400
Control Computer	1,600
Total	49,500

Precision

In order to measure the repeatability of positioning of the arm, a paper "target" with a millimeter scale was fixed to the last joint of the arm. The arm was stopped in a position in the center of the workspace. A laser pointer capable of producing a light point approximately 1 mm in diameter was fixed to point to the center of the target. The arm was then taken around a complicated path traversing and circling the workspace for approximately one minute. The arm was then sent back to the original position. To the precision of the scale and the observer's perception, the pointer was in the middle of the target. This was repeated for several different positions and angles, with the laser pointer mounted both horizontally and vertically, with the same results. The repeatability is therefore deemed to be better than ± 1 mm. The arm has also been tested to follow a straight path with sub-millimeter accuracy, but this has only been performed at very low speeds for safety reasons, so there are no experimental results for the accuracy at higher velocities.

Dynamic performance

The arm has been timed to traverse the operational window shown in Figure 5.2 vertically (distance 60 cm) in both directions within 0.39 s from standstill to standstill, which was predicted in the simulations. As for other movements — horizontal (60 cm) and diagonal (90 cm) traversion — only times of 0.5 s have been verified, as this is enough for our application and we want to minimize mechanical stress on the equipment. However, this implies that any point-to-point motion in the operational window takes at most 0.5 s to execute. The outermost joints are slightly slower then the inner ones, so the final angular alignment of the end effector rather than the positioning is the limiting factor for many configurations.

Cost Breakdown

The total cost of hardware used in the setup described here was just below 50 000 euros. For a detailed cost breakdown, see Table 5.7. Please note that these are the actual prices paid, and that there is no guarantee for future availability at these same prices.

5.2 User Interfaces

This section describes the various user interfaces used in the experiments.

Figure 5.5: Nest of Birds magnetic tracking device. From left to right: the magnetic transmitter, the control unit, and the markers.

Input Devices

This section describes the human input devices (HID) that were used for the experiments.

Nest of Birds Magnetic Tracker

For unconstrained free hand motion in the VR setting, we use a magnetic tracking device called Nets of Birds from Ascension Technology. This device can measure the position and orientation of up to four markers. Each marker is roughly a cube with 25 mm sides. According to the manual, the positioning accuracy is 1.8 mm, and 0.5 degrees. The update rate in our implementation is 100 Hz. The workspace is a sphere with 1.2 m radius centered on a magnetic transmitter, which is more than enough to measure free arm motions as long as the subject does not walk around.

We let the subject hold two markers in their dominant hand. The signals from the two markers are averaged in order to decrease measurement noise. The markers and the magnetic transmitter are shown in Figure 5.5.

Omega Haptic Device

For more conventional user input, we use a force feedback joystick from Force Dimension called the Omega. It is a parallel linkage device, with a large workspace, as well as high stiffness and force output. The position is sampled at approximately 2 kHz, and a Kalman filter is used to extract a velocity measurement. This gives very accurate, low-noise measurements of both position and velocity. This user interface hardware is shown in Figure 5.6.

Wiimote Inertial Game Controller

The wiimote user interface consists of a stock wiimote video game controller, see Figure 5.7. The features that are used in this setup are 3 linear accelerometers with a range up to ± 50 m/s^2 and 0.38 m/s^2 resolution, a 1024×768 pixel IR camera, and a trigger button.

Figure 5.6: Subject using operator station with Omega haptic device and stereo display. Photo by Mattias Bratt.

User input is read at 100 Hz. The accelerometers are reported to have an accuracy of $\pm 10\%$ [52], but this has not been verified in the present setup. The accelerometers were calibrated so that constant and linear bias errors were eliminated. However, noise when used hand-held is typically around 0.5 m/s^2, which is on the same order of magnitude as some users' input signals.

Figure 5.7: The Wiimote input device.

An IR LED setup provided with the wiimote is used with the camera to determine the orientation when the user is pointing forward. When the LEDs are not visible, an EKF keeps track of the orientation, also making use of the accelerometer data generated by gravity whenever the wiimote is assumed to be nearly motionless. When the LEDs are not visible, the orientation measurements become slightly less accurate, but the main difference to using LED data is that rotation can not be measured along the vertical axis, limiting tracking to 5 DoF instead of 6, assuming that there is no rotation around the vertical axis. Apart from tracking the direction the user is pointing, the orientation data is also used to subtract the Earth's gravity from the accelerometer readings, as well as to convert measurements from the wiimote frame of reference to the global frame of reference. The trigger button is connected to a reset function that sets position and velocity to zero, used to initialize the reference zero position.

5.3 Sensors

This section describes the external sensors used.We employ a vision system consisting of a wall-mounted stereo camera pair from Videre Design with a 60 cm baseline, see Fig 5.8. The cameras connect via Firewire to a Dell Precision 9510 workstation with a Pentium D dual core processor at 2.8 GHz. This setup allows for stereo pairs of color images taken at 50 Hz at a resolution of 320×240 pixels. The cameras have a field of view of approximately 50^o in the horizontal plane.

Figure 5.8: 60 cm baseline stereo cameras

For ball catching experiments, these cameras are used to track the balls, using an extended Kalman filter (EKF), as described in [44]. The ball is detected in each image using simple color segmentation. First, the 24 bit RGB image is converted to 24 bit HSV using a lookup table. The ball was found to have a *hue* value of 3, and a (largely varying) *saturation* value of approximately 160, so all pixels that are in the range 1–5 for hue and 120–200 for saturation are preliminary marked as ball pixels. A second pass that only keeps marked pixels with at least 3 other marked neighbors eliminates noise. The center of mass for the marked pixels is calculated and used as the ball centroid. A subwindowing scheme is applied, since these have shown to be very efficient to significantly reduce the time needed in segmenting moving objects (c.f. [73]). After the ball has been detected the first time, only

a subwindow where the ball should be expected to be found is processed. This subwindow is calculated using the state estimate from the EKF, and the size of the window is set to cover several times the standard deviation in position. Using this approach, the ball can be segmented and localized with a reasonable accuracy at less than 4 ms processing time per stereo image pair, giving sufficient real-time performance.

For other motion capture purposes, such as tracking the absolute position of the wiimote, the cameras are used to track a small (5 mm $\times$ 5 mm) plastic marker of the same color as the balls, using mostly the same approach. The main difference is that for general motion capture, no assumptions can be made about the motion of the marker, so that instead of using an EKF, the 3D position is just recorded as is. Thus, there is no estimated standard deviation available to decide the subwindow size, but this is instead set to a fixed size of 60×60 pixels centered around the last tracked position, which showed empirically to work well, due to the tracked motions being much slower than those of the flying ball.

Displays

This section describes the different displays used for presenting feedback to the user.

Stereoscopic Monitor

A stereo display was realized with a Philips Brilliance 109P4 monitor and shutter glasses. The monitor updates the image at 120 Hz, meaning that each eye sees a relatively flicker-free image updated at 60 Hz. The visualization of the remote site is performed using Open Inventor 6 from Mercury Computer Systems.

Even though the Omega device setup and the software developed supports feeding back robot arm momentum and obstacle avoidance forces to the user, this functionality was disabled for the catching experiments. Instead, the force control in the device was only used to enforce the limits of the workspace and to nullify gravity and friction.

The user interface receives the estimated position of the ball from the stereo vision system of Section 5.3. To be able to bridge communication time delays using prediction, it also needs the complete state of the Kalman filter to make future ball position estimates. Estimates are computed by letting the state develop further forward in time without the usual measurement update. This predictive power is also used over the much longer time interval of a complete throw to generate a projected trajectory of the ball. This is shown as a red line through the ball, and the uncertainty inherent in the future Kalman states is displayed as a semi-transparent shell around it. The shell cross section that corresponds to a particular point on the trajectory is the projection along the predicted velocity of an uncertainty ellipsoid. The axes of the ellipsoid are twice the standard deviations along different directions derived from the eigenvectors and eigenvalues of the Kalman filter covariance matrix for the predicted x, y, and z coordinates of the ball. The result is that the uncertainty of the ball trajectory drawn is visible as a funnel that has its narrow end at the current ball position, and widens as the standard deviation, represented by the distance from the trajectory in the center, increases the further into the future the prediction extends. Screen dumps of the operator interface are shown in Figures 5.9 and 5.10.

When the ball is not in flight, but is visible to the vision system, the user interface computer still gets information about the estimated ball position. In this case, which often occurs just before a throw, the rest of the Kalman state is not available, and the ball is shown in an alternate color. As the vision system determines that a throw has begun, full Kalman state data starts to flow in, and the word ‘go’ is output to the speaker to alert the user. Another sound is played if and when the ball is caught.

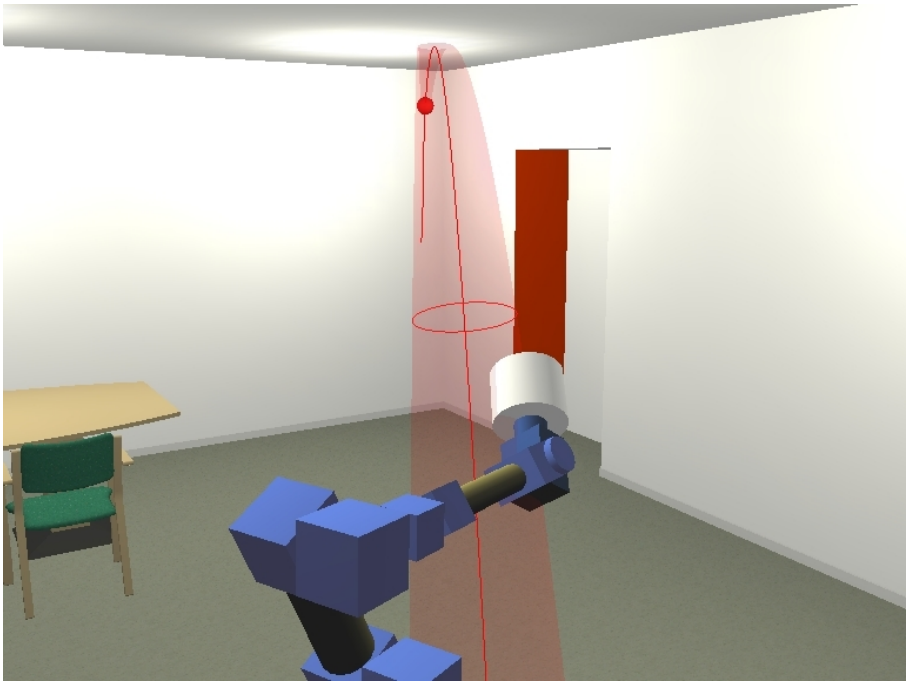


Figure 5.9: The user's view, as taken from the ball-catching experiment.

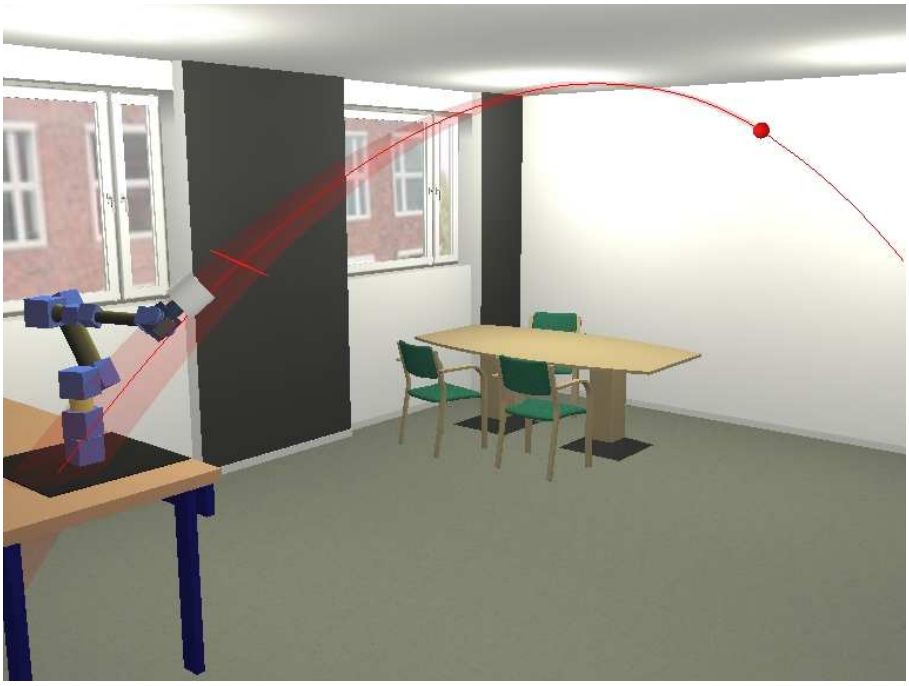


Figure 5.10: A side view of the virtual world of the ball-catching experiment.

All data about user actions, as well as received robot and ball states, are logged to file by the user interface computer. The log files can then be exported and used for data analysis, but also serves as input to the interface software in replay mode. This mode of operation allows normal and reduced speed play-back and pausing of recorded experiment sessions. The viewpoint and viewing direction can be arbitrarily adjusted in real-time using a 3Dconnexion SpaceTraveler six DoF mouse-like input device.

Head Mounted Display

For the head mounted display (HMD), the stereo image is displayed on an eMagin 3DVisor Z800, aimed at the high-end gaming market. It has one 600×800 OLED display per eye, and a diagonal field of view of 40 degrees. For headtracking and tracking of free hand motion the Ascension Technology Nest of Birds device described in Section 5.2 is employed. To get acceptable quality headtracking we use a setup with two of the four available markers fixed to the user's head and the magnetic transmitter unit mounted at a distance of only approximately 30 centimeters from the markers. This allows measuring the orientation and position of the head with the in-device filtering disabled. It otherwise causes unacceptable lag and, consequently, nauseates the user.

USB Camera Video Stream

A direct video stream of the remote site is realized with an inexpensive USB camera with 640 × 480 resolution and a framerate of 15 fps, which was mounted to point towards the robots workspace. The unprocessed video image is then shown on a 19-inch Samsung SyncMaster 940x LCD monitor using xawtv software, c.f. Figure 5.11.

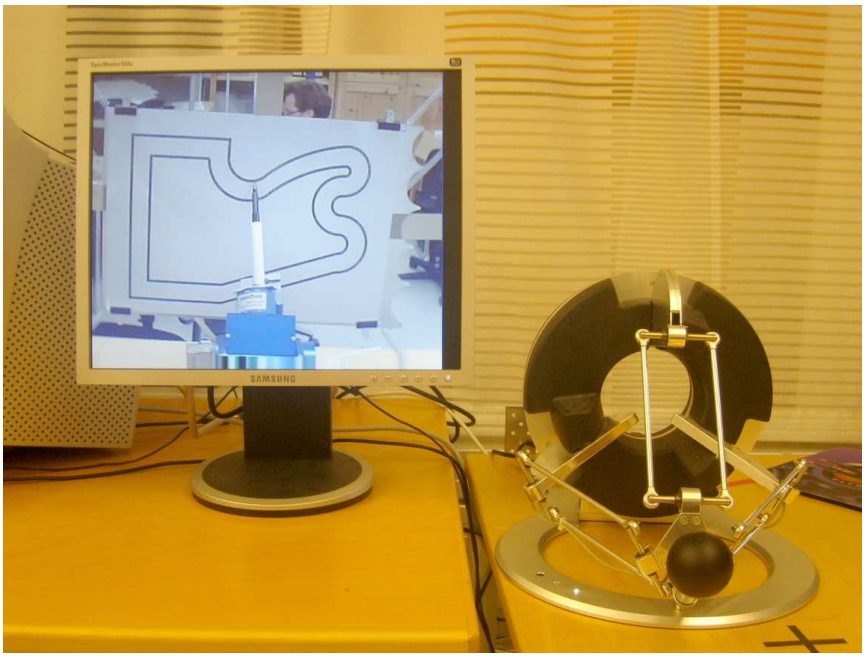


Figure 5.11: User interface hardware, with video display of remote site and haptic input device.

5.4 Communication Handling

For teleoperation purposes, and to simplify changing the operator interfaces and introduce time delays, all parts of the system are connected over an IP network, either locally on the LAN in the lab, or globally using the Internet. Different kinds of fluctuations in the delay include random noise, changes due to variable network load, delays from transmitting too much data, bunching of packets not originally transmitted together, and lost packets [109].

For transmission of real-time data UDP (user datagram protocol) is preferable to TCP (transport control protocol), because it avoids the overhead of e.g. detecting and retransmitting lost packets, at the expense of not guaranteeing transmission of all data or constant ordering [105]. If the packets are self contained, i.e. they each contain one sample of all transmitted signals, only the newest packet available at the receiving side is relevant, making retransmission of old data and enforcing constant ordering unnecessary.

We perform a simple experiment, repeating that presented in [105], to illustrate the difference between UDP and TCP. For the test to be comparable, the exact same transmission characteristics are used. We sample transmission characteristics between the Royal Institute of Technology (KTH) in Stockholm, Sweden and the University of Genova (UGE) in Genova, Italy. The samples were taken at 10:40 in the morning on September 25, 2008. The minimum delay was 46 ms, and the median delay was 48 ms. A histogram of the delays is shown in Figure 5.12. We then transmit a simple cosine signal between two computers connected on the same LAN, with no significant transmission delays, but delay each packet according to the sampled real internet delays. For TCP, all packets arrive, and the ordering is preserved. For UDP, we use time stamps to be able to drop all arriving packets that arrive out of order, i.e. after a packet with a later time stamp. This means that with UDP, some packets will be dropped, but the overall real-time performance is better, see Figure 5.13.

All separate parts of the system have synchronized clocks, so that time stamps set by different parts of the system can be directly compared, and delays are easy to measure.

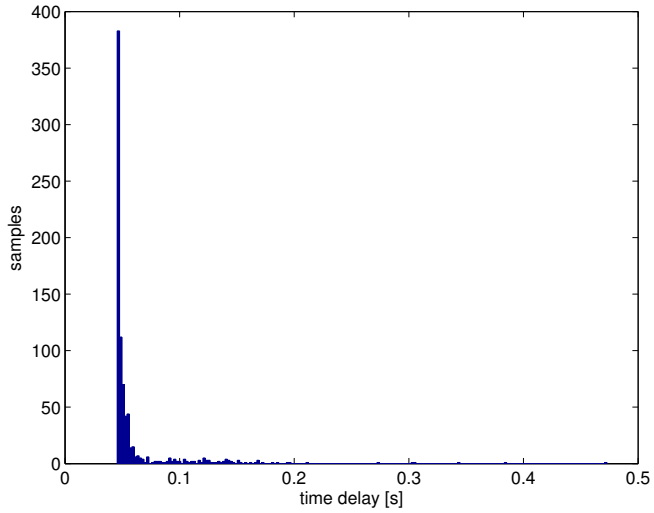
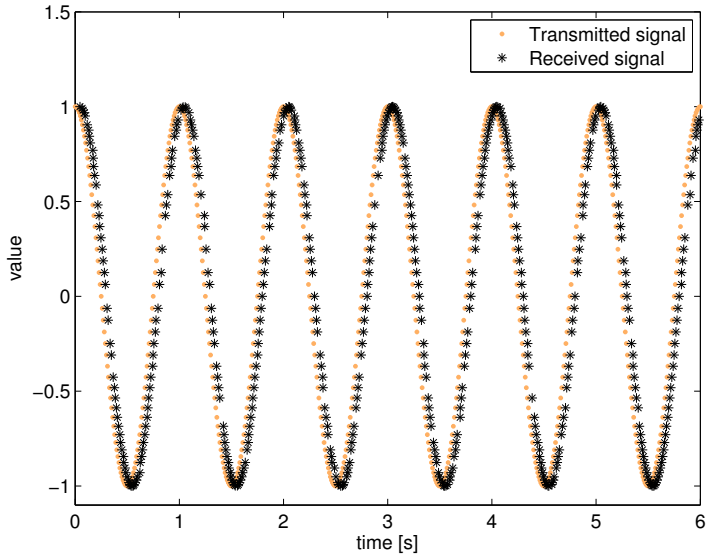
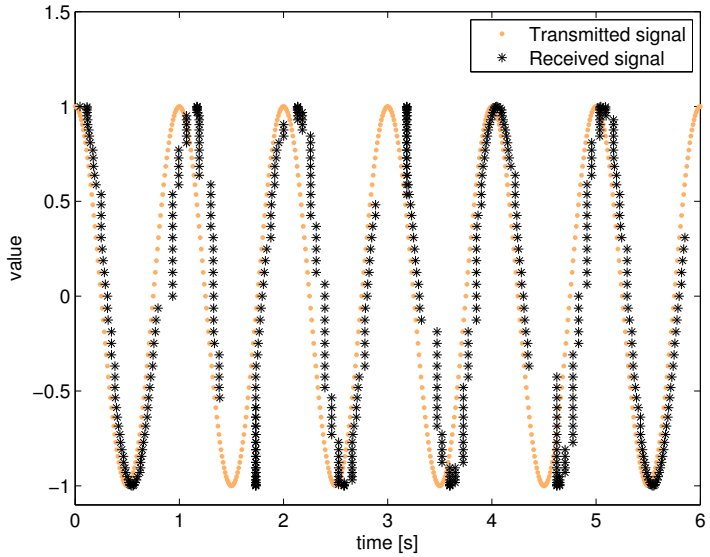


Figure 5.12: Histogram of internet delays between KTH and UGE at 10:40 CET on Sep. 25 2008.



(a) Comparison of received and transmitted signal when using UDP over an internet connection with stochastic delays. Some packets are allowed to be dropped, but doing so allows the remaining packets to have be delayed less.



(b) Comparison of received and transmitted signal when using TCP over an internet connection with stochastic delays. No packets are dropped, but enforcing this causes delays in some of the subsequent packets.

Figure 5.13: Comparison of transmission characteristics for UDP and TCP.

Chapter 6

Offline Experiments

The proposed hardware setup, experimental scenarios, and minimum jerk (MJ) motion models were given a first evaluation in a series of experiments. The common factor in these experiments is that no part of the robot control depends on the validity of the models. All analysis is made on recorded signals after the experiments have been carried out. The results gained from this approach will then be used to design and implement on-line experimental setups.

6.1 Ballcatching with Simulated Robot

The first experiment is an initial pilot study of the two 3D user interfaces, to determine if they are appropriate for the ballcatching task, and to see if the minimuim jerk (MJ) model is applicable to estimate user input when using the interfaces for this task. In the experiment, we want to accomplish several goals. First, we want to evaluate the experimental setup, to determine the task difficulty and examine how subjects interact with the UIs. We also want to collect user input data to get an initial indication of what kind of input signals to expect.

We want to compare the 3D Monitor + Haptic UI (setup 1) with the VR HMD UI (setup 2), both described in Section 5.2. The initial assumption is that setup 2 should allow for interaction that is natural enough that the MJ models are valid. We want to compare this to the results from setup 1, to determine if MJ models are also valid for the latter.

In the user trials, only the simulated virtual reality system was used. Since the goal was to study the interaction between the user and the interface, it was deemed that connecting the actual robot arm would not benefit the experiments. In the setup, the same virtual world was presented in the two different interfaces. The user's viewpoint was slightly behind and above the manipulator with a clear view of the rest of the room and the path of the incoming balls, see Figure 5.9. Balls were visualized as small spheres.

Prior to the experiment, a ball was thrown several times to the robot described in Section 5.1, which was set to autonomously catch any balls by intercepting the predicted ballistic trajectory. The ball trajectories of 21 successful catches were recorded, verifying that all trajectories were physically possible to catch. The recorded ball trajectories could be replayed at will, and in the same order for all experiments.

The subjects were given a brief explanation of how the interface works. They were told that they would see balls being thrown towards them, and that they were to try to steer the robot arm so as to intercept the ballpath and thereby catch the ball. With setup 1, the subjects were instructed that the arm would copy the movements of the input device,

Table 6.1: Catching success rates

Subject	Setup 1	Setup 2
1	0.62	0.71
2	0.24	0.33
3	0.29	0.38
4	0.48	0.67
5	0.48	0.52
6	0.29	0.10
7	0.76	0.71
8	0.81	0.62
9	0.71	0.33
10	0.33	0.43
total	0.50	0.48

and were given some time to explore this before the ball-catching task was introduced. In setup 2, subjects were told that the robot arm would mimic the motions of their right hand (or left for the one left-handed subject), and that the end effector of the manipulator would be visualized as being where they would normally expect to see their hand.

For each system, the subjects were given 20 practice throws, in which the balls were thrown in a random order. After a short break, they were subject to 21 throws that they were told would be measured. When using setup 2, the subjects were allowed a short rest after each 7 throws, dividing the experiment into three shorter sessions. This was so that they could rest their arms/shoulders in order to minimize fatigue. The setup may cause users to stand with their arm in an outstretched position that causes strain on the shoulder, and if not explicitely instructed to rest subjects may concentrate too much on the robot control task to notice.

A total of 10 subjects were used, all of whom are robotics or computer vision researchers affiliated with our university, but with no prior knowledge of the setup. Of these 3 were female and 7 were male. The same subjects were used for both setups, but the trials with the different systems were conducted approximately two weeks apart. The trials with setup 1 were performed before the trials with setup 2.

Results

In the simulation experiments, the catch performance varied largely between subjects. Some subjects were able to successfully catch most balls after the initial practice, while others were hardly able to catch any. The success rate for the different setups are shown in Table 6.1. When making comparisons with the performance of the real robot setup in later experiments, it should be noted that balls are much easier to catch in the simulated environment, mostly due to the lack of modelling of collision dynamics. Thus, all balls that hit the catching cylinder are considered to be “caught” in the simulator, while many of these might bounce off in the real system.

Although not explicitely studied here, debriefing with the subjects hints at a correlation between successful stereo fusion in the interface and successful catching. some subjects with poor results complained that they did not feel adequately “immersed” in the interface. It is more difficult to intercept a ball trajectory in three dimensions for a user that can only perceive two.

Table 6.2: Average times and distances for MJ motions on setup 1

Subject	Avg Δt [s]	Std(Δt) [s]	Avg distance [m]	Std [m]
1	0.3376	0.0859	0.2115	0.0859
2	0.2974	0.0462	0.5753	0.1078
3	0.3906	0.0498	0.3638	0.1216
4	0.3076	0.0940	0.1806	0.0692
5	0.3883	0.0853	0.2594	0.0678
6	0.3059	0.0948	0.3552	0.1283
7	0.3190	0.0839	0.1945	0.0724
8	0.2874	0.0863	0.1935	0.0565
9	0.3738	0.0876	0.4557	0.0731
10	0.3471	0.1015	0.2348	0.1382
total	0.3355	0.0801	0.3024	0.1571

Table 6.3: Average times for MJ motions on setup 2.

Subject	Avg Δt [s]	Std(Δt) [s]	Avg distance [m]	Std [m]
1	0.7229	0.1063	0.2579	0.1174
2	0.5268	0.0628	0.5082	0.1018
3	0.6738	0.1430	0.3190	0.1832
4	0.6799	0.1291	0.3987	0.2876
5	0.8269	0.1361	0.3135	0.0896
6	0.4860	0.0965	0.4211	0.1330
7	0.6703	0.1162	0.3315	0.1264
8	0.7474	0.1201	0.3971	0.0977
9	0.5945	0.1010	0.3774	0.1282
10	0.6112	0.1218	0.5137	0.1574
total	0.6542	0.1492	0.3838	0.1688

The average time and average traveled distance for the the first MJ trajectory for the different subjects using setup 1 are presented in Table 6.2. The distances are measured in the robot space for easy comparison. The actual distance moved with the device is 10 times smaller. Given that all subjects try to catch balls from an identical set of trajectories, it could be expected that the differences in traveled distance would be smaller. Some subjects would return the robot to a default position between catches. This position varies across subjects, and accounts for most of the difference in average distance. The average time and the average motion distances (in robot space) for the first MJ trajectory using setup 2 is presented in Table 6.3. We see that the distances moved with the robot and the variations of these do not vary much between the setups, as is expected as the subjects are trying to perform the same task in both setups. However, there is a large difference in the motion time, with subjects taking almost twice as long to perform the motion in setup 2. A probable explanation for this is that setup 1 has a scaling factor of 10, thereby allowing the subject to move up to 10 times faster.

Minimum Jerk Fitting

We examine how the recorded motions from the experiment can be approximated with MJ trajectories. As the subjects were given several tries to practice, we also assume that the initial reaching motion will be of approximately equal duration in different trials with the same person, as the velocity of the approaching ball and the distance to the intercept point is similar between trials. Our first approach was therefore to extract the average duration $\Delta t = t_1 - t_0$ for a reaching motion.

In order to find Δt, it is important to identify where one MJ trajectory starts and ends. Since the total motion can contain several MJ trajectories superimposed on each other, they must be separated. Our method for separation works by identifying the typical bell-shaped tangential velocity profiles of a MJ motion [100, 41]. These bell-shaped velocity profiles are 4th-degree polynomials with boundary value conditions that state that they start and end at zero, with zero derivatives, c.f. 3.2. This leaves only one free parameter, which can be estimated using a least-squares approach. Since we anticipate several superimposed trajectories, we fit several such 4th-degree curves, minimizing the mean square error of the sum of these trajectories as they approximate the total velocity profile. An example of the results of such a fitting is shown in Figure 6.1. We assume that submovements are at least 100 ms apart, as observed by Milner [100].

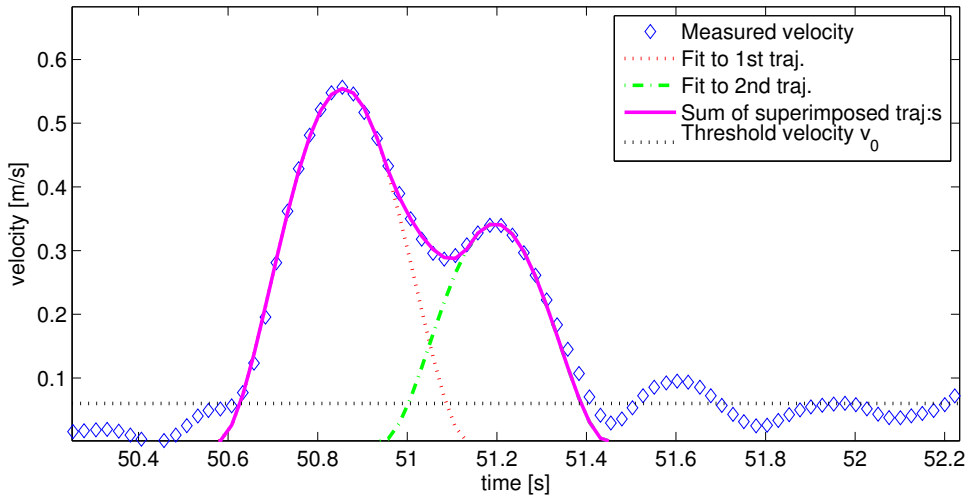


Figure 6.1: A fitting of two superimposed 4th-degree polynomials to a measured tangential velocity profile

We extract the average duration of the first motion of each recording, and use for Δt. The parameter t_0 was determined by finding the point in time after a throw commenced that the motion exceeded a predefined threshold velocity v_0. We fit Equation 3.2 to the data using least squares fitting.

We also want to see if MJ models are appropriate for predicting future motion. We do this by performing the least squares fit using points in the interval from t_0 to $t_0 + \alpha\Delta t$, where $\alpha \in [0,\ 1]$ is the portion of the motion used for fitting. The less data that is used for fitting the MJ trajectory, the earlier a prediction can be made. However, it is expected that the quality of the prediction will deteriorate with the use of fewer samples. It is therefore

of interest to find what performance can be expected when using different portions, α, of the trajectory data before making a prediction.

To illustrate the fitting of a MJ trajectory to recorded data, two examples are shown. Both examples originate from subject 1, for whom the performance of MJ fitting was average on setup 1, as shown in Figure 6.3.

The first example, in the left part of Figure 6.2, illustrates a catching motion that was close to the MJ model, and could be fit well to a MJ trajectory. The plot shows the tangential velocity profile at the top, and the different components below. The green dotted line is the fit MJ trajectory, ended with an asterisk. The first vertical line represents the start of the motion, and the second vertical line shows the last sample used for fitting. In this example, a portion of $\alpha = 0.35$ of the motion was used for prediction. The second example, in the right part of Figure 6.2, shows the performance when a large corrective motion is added after the initial motion. As can be seen, this motion is not a good fit to the MJ trajectory, mostly due to the fact that almost no samples of the corrective motion are used for fitting the MJ trajectory.

The average deviation from final position for the fitted MJ trajectories for all subjects is shown in Figure 6.3. This was calculated by using the samples from the first portion of the motion to fit an MJ trajectory, and then measure the distance between the end of the predicted trajectory, and the actual position at this time. When only a small portion ($\alpha < 0.2$) of the samples were used, insufficient data accounts for the larger part of this deviation. When more samples are used, non-modeled corrective movements after the used samples account for most of the deviation.

To evaluate the performance of MJ trajectory fitting, the results are compared to other, simpler models. For MJ fitting to be meaningful, it should at least outperform simpler approaches that do not use human motion models. Perhaps the simplest fitting possible is to use the last available sample and assume that the subsequent signal remains constant.

Since this model does not assume any motion after the last observation, it can be viewed as a zero-order model. If motion is assumed, using least square fitting to fit linear, quadratic, or cubic functions to the data set can be used. A comparison of the performance of MJ trajectories and extrapolation of polynomials of different degrees using setup 1 is shown in Figure 6.4. The average deviation from the final position for the fitted MJ trajectories using setup 2 is shown in Figure 6.5. A comparison of the performance of MJ trajectories and extrapolation of polynomials of different degrees is shown in Figure 6.6.

Polynomials of higher degree than 3 result in even larger deviations when used for extrapolation. Note that in all these fittings, the temporal duration of the motion, i.e. determining when to stop, was found by fitting the MJ model to the velocity profile as described above. Since the MJ model in itself is a 5th degree polynomial, the difference in predictive performance is explained by the boundary conditions.

If we pick out just the motions where there is only one major MJ component, the predictive performance of the MJ model for these motions is much better. For these tries, which make up 60% of the total tries, the prediction error is less than 0.07 m (the radius of the ball catching end effector) after 40% of the motion has passed. We can also observe that the error of the MJ-based model fit does not reach zero even as α reaches 1. This is due to the actual recorded motion not being a perfect single submotion MJ trajectory.

Retry With Expert Subject

As shown in the results above, the performance varies significantly between subjects. Also, we see that subjects that perform better at catching fit the MJ model better. We therefore decided to isolate one subject and let him have a large amount of training with the system,

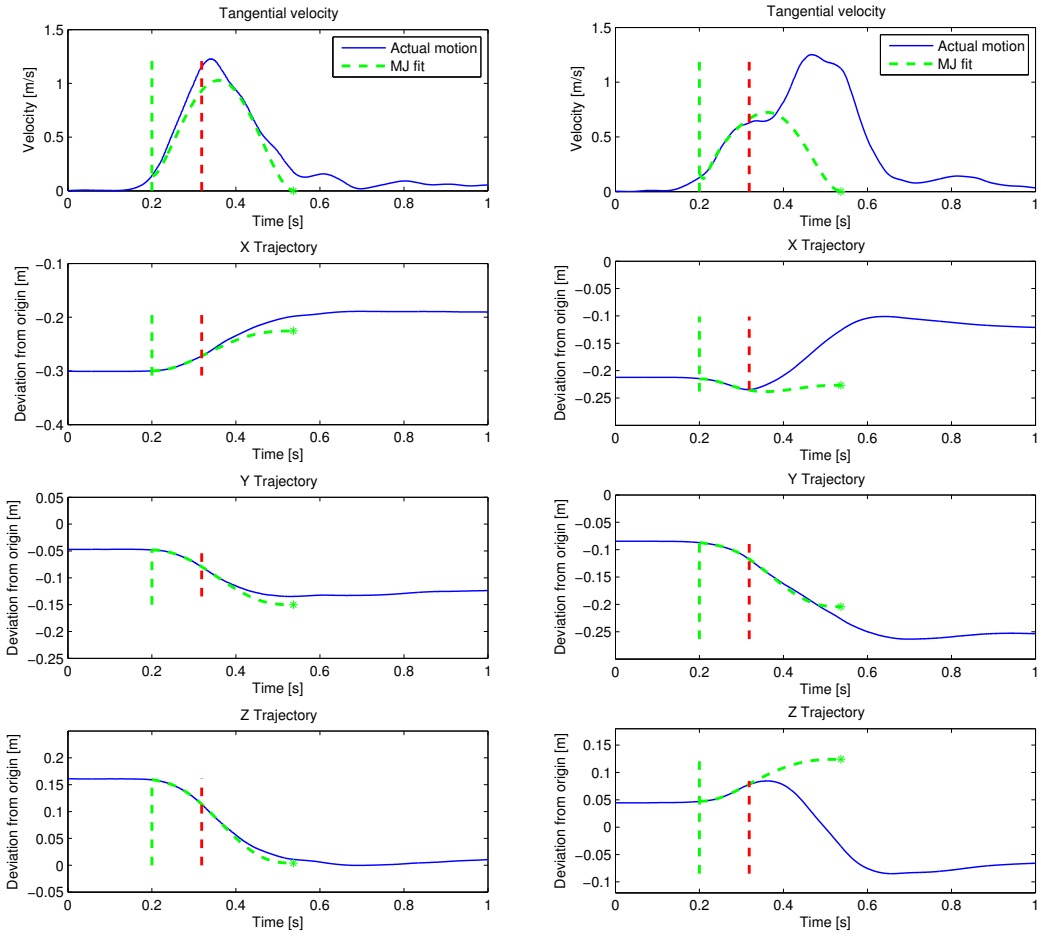


Figure 6.2: Examples of fitting a MJ trajectory to a catching motion using data from the first 35% of the motion to fit. The left plots show a successful fit, and the right plots show a failed fit. The solid line is the measured trajectory, the dotted line is the MJ trajectory fit. The two vertical dashed bars enclose the portion of the trajectory used for fitting. The portion of the MJ trajectory that continues beyond this portion is purely extrapolated.

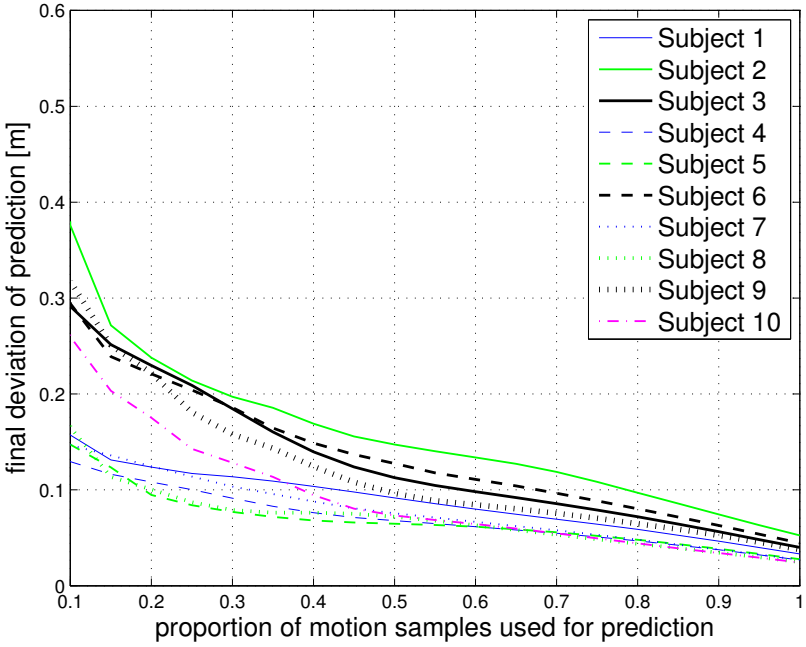


Figure 6.3: The final deviation of the predicted MJ trajectory as a function of the portion of the trajectory used for fitting, using setup 1.

eventually reaching a performance of almost 100% successful catches. We then let this subject perform the experiment with setup 1 and examine the estimation performance. In order to achieve reasonably statistical significance, we let this subject perform 5 times as many tries as in the previous experiment, or a total of 105 tries. Of these, all 105 tries were successfully caught. With this experiment, we isolate user input that is well described by the MJ model, and should thereby acquire an upper limit of the potential performance of the approach.

Some tries that did not generate interesting data are excluded from parts of the analysis. For example, for some tries, by chance the subject was in the correct final position before the throw began. For these tries, a successful catch was performed without moving the robot at all, and the prediction of the motion endpoint could trivially be found within a very high absolute accuracy. There were approximately 12 tries where the initial position was too close. Also, for some throws, the operator made a very poor initial motion and the subsequent corrective motions were of equal or larger magnitude than the first. There were approximately 24 throws where the operator's initial guess was significantly off. This left 69 throws to be considered in the main part of the analysis. The performance on all recorded motions is also presented, for reference.

The data collected in the experiment was analyzed offline, and the endpoint of the hand motion was continually estimated as time approached the catching instant. This prediction was then compared to the hand catch position to test the quality of the prediction. The results of this analysis are shown by the plots in Figures 6.7 and 6.8.

The plots demonstrate that for most hand trajectories in the selected set, good estimates of the catch position become available during an interval from approximately 250 to 200 ms

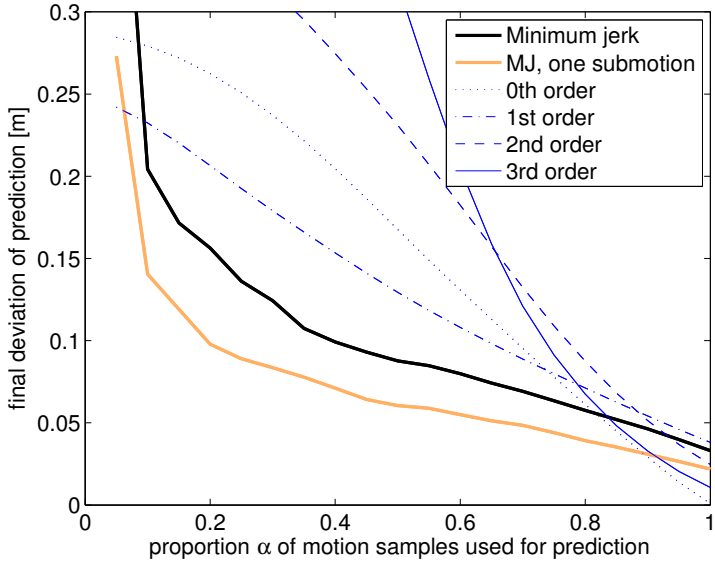


Figure 6.4: The final deviation of the predicted MJ trajectory as a function of the portion α of the trajectory used for fitting, as compared to the final deviation when fitting polynomials, using setup 1. This is the average over all subjects. The line labeled 'one submotion' represents the tries where only one significant MJ submotion was identified.

before impact. At around 200 ms the time advantage before the same level of convergence is reached without prediction is approximately 70 ms.

The threshold distance of 6 cm from the catch position that was used to generate Figure 6.7 was chosen by examining several similar plots. Increasing the threshold slightly beyond 6 cm gives little improvement in terms of earlier convergence, whereas decreasing it causes a substantially larger delay. A sphere with radius 6 cm has a volume equal to 2.3% of the smallest possible convex volume containing all the hand catch positions. Also, given that the accuracy of the ball tracking setup is roughly ± 1 cm, a 6 cm radius in prediction should guarantee that the total accuracy covers the 7 cm radius of the ball-catching end effector, thus resulting in a successful catch in reality.

Observations and Conclusions

One of the most important observations we can make in this experiment is that the MJ model seems adequate for both tested types of user interface. Somewhat surprisingly, the MJ model gives better early predictions for input via the haptic interface than for input via the VR interface. The simple explanation for this is probably that setup 2 has a significantly lower sampling frequency than setup 1, and therefore the results for using only the first 10 or 20% of the samples are not very reliable. With 100 Hz sampling and a 0.6 s motion duration, the first 10% of a motion consist of only 6 points, which will not result in stable fitting to a 5th degree polynomial.

As can be seen from the plots of fitting performance, the MJ fitting outperforms the other fitting methods, at least for a portion of the interval. In both experimental setups it seems that MJ fitting is better when one to two thirds of the data is used for fitting.

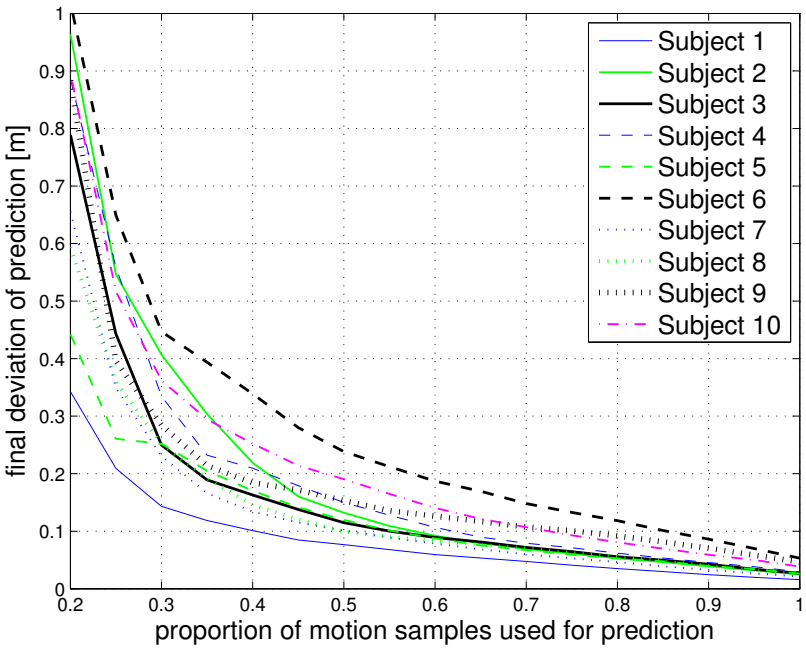


Figure 6.5: The final deviation of the predicted MJ trajectory as a function of the portion α of the trajectory used for fitting, using setup 2.

However, it is also obvious that this performance varies greatly with the subject. There seems to be a connection between successful catching and successful fitting to a MJ trajectory. An explanation for this can be that subjects that perform well with the catching task are able to make a good early prediction of the ball trajectory and therefore need less corrective motions. This proposition is supported by the observation that many of these subjects tend to move shorter distances with less variation, as can be seen in Tables 6.2 and 6.3. Accordingly, when we enlist an expert subject, the performance of the predictor is significantly improved.

It should also be noted here, that the average deviation does not give all information. As illustrated in 6.2, if the motion contains only one major MJ component, the prediction error is much smaller than if the motion also contains significant corrective submotions. We could therefore expect reasonable predictive performance for such user input that only contains one major MJ submotion.

Another observation is that there is a large difference in where the subjects start their motions, some start close to the robot center position, and do not have to move very far to catch a ball, while some start far from the center position, and have to move farther to reach the catch position. The distribution of motion distances is shown in Tables 6.2 and 6.3. This difference in moved distance may give different users different difficulty levels, and should be corrected in the online experiments.

From this, we conclude that setup 1 allows interaction that is close enough to unconstrained free motion that the MJ model is applicable. We also see that with this setup, the largest accuracy gain for prediction mith MJ models as compared to other approaches is found when approximately one third of the motion has been executed.

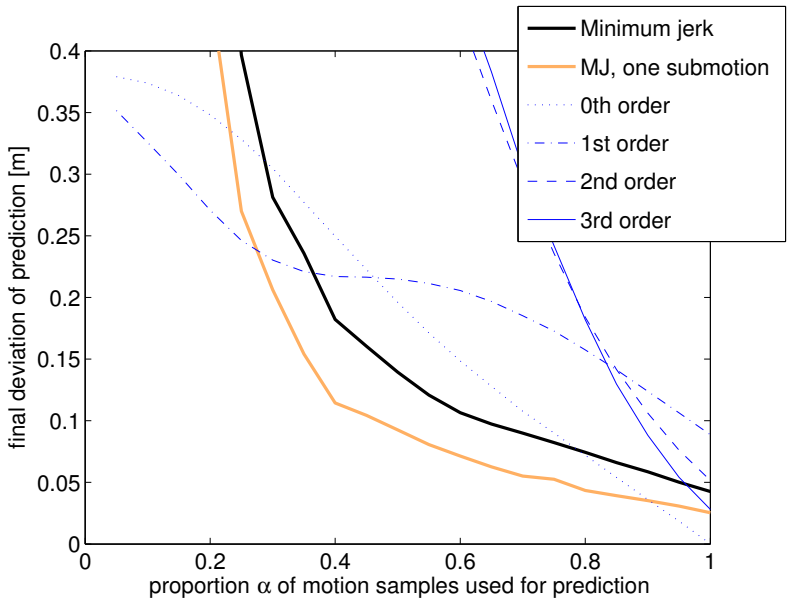


Figure 6.6: The final deviation of the predicted MJ trajectory as a function of the portion α of the trajectory used for fitting, as compared to the final deviation when fitting polynomials, using setup 2. This is the average over all subjects. The line labeled 'one submotion' represents the tries where only one significant MJ submotion was identified.

6.2 Wiimote Tracking

An offline tracking experiment was conducted using the wiimote interface. The purpose of this experiment is to record motion data and examine the quality of the input signals from the wiimote controller. We also want to determine if position tracking is possible, and if it can be enhanced by applying MJ models. This is done with a comparative analysis of MJ model based tracking, a naive integration approach, and state of the art target tracking using a Kalman Filter.

To define the motions for data collection, a set of physical targets were set up. These consist of colored plastic balls that can be touched with the wiimote. One ball was designated as a starting point, and three other balls were mounted on a line 0.3 m above it, see Figure 6.9. Subjects were asked to start by touching the starting point ball, and then touch the colored balls as called out by the experimenter. In essence, this is a visually guided reaching motion - the type of motion that MJ models are explicitely claimed to describe.

The starting point was located at about chest height in front of the subject, and the other targets were located approximately at face height. Each subject was asked to carry out 40 such touching motions, according to a predetermined pattern.

In total, 5 subjects were used. They were not given any other instructions than to start stationary and touch the colored balls as called out, and had not been told the purpose of the experiment or the details of the tracking system before the experiment. The motivation for this was that this user interface should be intuitive to use, and we want to see how well we can track motions of uninformed subjects. This is of interest for potential future applications.

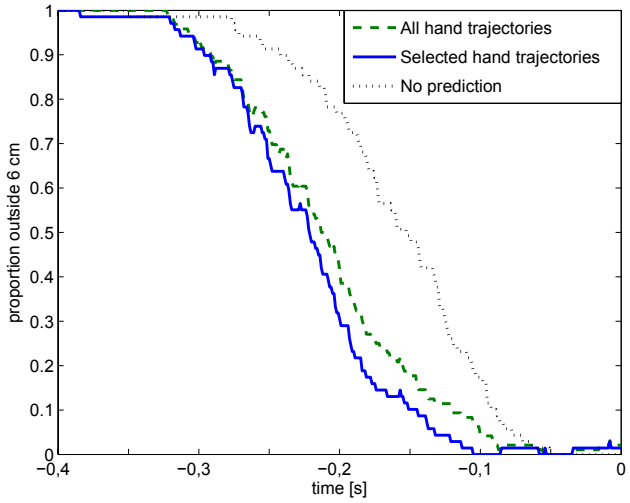


Figure 6.7: Proportion of predicted hand end positions outside 6 cm radius from actual hand catch position. Results are shown for selected hand trajectories, all hand trajectories, and for the case of using current hand position instead of a predicted endpoint (there is no visible difference when the latter curve is plotted for all hand trajectories or just the selected ones). For a fair comparison the curve representing the complete set of hand trajectories has been modified as to not include nine trajectories that are completely contained within the 6 cm radius.

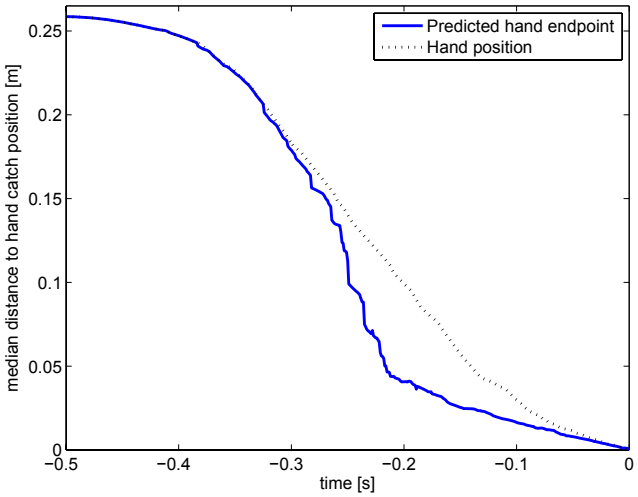


Figure 6.8: Median deviation of the predicted endpoints for selected hand trajectories catches from actual hand catch position.

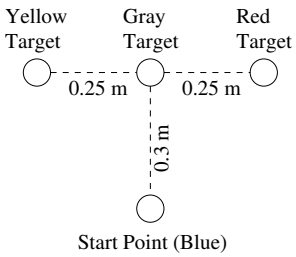


Figure 6.9: Schematic of targets used in tracking experiment. Subjects start with the wiimote touching the start point, and move the wiimote to touch the colored targets as called out by the experimenter.

Tracking Problem

The wiimote uses inexpensive MEMS accelerometers. Consequently, the accuracy is limited and one has to consider the challenges of using sensors that have limited accuracy, and there are two main problems one will face when attempting to track position.

The first challenge is one of observability. In the strict sense of the word, position is not observable from acceleration measurements alone, as one will not know the initial position or velocity. In practice, however, one can assume that a motion starts at rest at the origin, adding initial conditions that resolve the problem. The second problem is more difficult. A measurement $\hat{\ddot{x}}$ of acceleration $\ddot{x}$ will be corrupted by noise w:

$$\hat{\ddot{x}} \sim \ddot{x} + w \tag{6.1}$$

Thus, when we integrate $\hat{\ddot{x}}$ twice to get an estimate $\hat{x}$ of position x, we will have that

$$\hat{x} \sim x + wt^2 \tag{6.2}$$

That is, the error is now proportional to t^2, and will tend to infinity over time. This is a well-known problem, and has been partially solved for shorter durations by removing bias errors and filtering the results, with e.g. a Kalman filter [11, 94]. However, these approaches only minimize the error coefficient w, they do not eliminate the t^2 dependency. Due to these problems, up to now, apart from finding the general direction of the gravity vector, low-resolution accelerometers have mainly been used for qualitative input, such as gesture recognition based interfaces [80, 145].

In practical cases, just taking the naïve double integral of acceleration measurements from a wiimote will give position estimates where the noise to signal ratios exceed 1 after just a few seconds, given the noise magnitude of approximately 0.5 m/s^2.

Minimum Jerk Fitting

For this experiment, using the assumption that the user only performs isolated reaching motions, an input tracking system based on MJ theory was designed. Here, "isolated" is used in the sense that one motion does not start until the previous motion is ended, in practice requiring a 100 ms separation. Thus, the constraints are set so that $\dot{x}_0$, $\ddot{x}_0$, $\dot{x}_1$, and $\ddot{x}_1$ are all zero, and each motion is assumed to start at the postition where the last one ended. This ensures that all motion estimates end at zero velocity, eliminating buildup error. Since the measurements are in acceleration space, Equation 3.1 was differentiated twice to get Equation 6.3.

$$\ddot{x}(t) \;=\; 20a_1t^3 \;+\; 12a_2t^2 \;+\; 6a_3t \;+\; 2a_4 \tag{6.3}$$

We assume that all motions start and end with zero acceleration, which gives the following constraints:

$$\ddot{x}(t_0) \;=\; 0, \qquad \ddot{x}(t_1) \;=\; 0$$

Then, Equation 6.3 can be rewritten parametrized by the single constant a_1, as in Equation 6.4.

$$\begin{aligned} \ddot{x}(t) \;=\;\; & a_1(20t^3 - 30(t_0 + t_1)t^2 \\ & +10(t_0^2 + t_1^2 + 4t_0t_1)t \\ & -10(t_0^2t_1 + t_0t_1^2)) \end{aligned} \tag{6.4}$$

Given a consecutive stream of acceleration measurements, Equation 6.4 can be fitted to a sequence of these using least squares fitting. The initial state is assumed to be non-moving, and all acceleration measurements beneath a certain threshold are ignored. When four consecutive measurements above the threshold have been recorded, the fitting process is initialized. Thus, in the beginning of a motion, only a small part of the curve will be fit to actual measured points — the rest will be extrapolated.

Two of the parameters in Equation 6.4 are the unknown start and end times of the motion, t_0 and t_1. In a first attempt, we tried to find the correct parameters for the MJ model using a particle filter. However, as exploratory experiments showed that almost all hypothesis had to be kept for long part of the motion, this probabilistic approach was abandoned in favor of finding t_0 and t_1 using an exhaustive search, which performed more consistently with only a slightly higher computational cost.

Results from 6.1 imply that we should expect motion durations from 0.4 s to 1.0 s, so we search an interval from 0.3 s to 1.3 s to cover all expected durations. We call the time where the acceleration threshold is exceeded t_{tr}. We then let t_0 take all values from $t_{tr} - 100$ ms to $t_{tr} + 100$ ms in 10 ms steps, and for each value of y_0, we let t_1 take the values from $t_0 + 300$ ms to $t_0 + 1000$ ms, also in 10 ms intervals. For each interval $[t_0, t_1]$, we find the least square fit using all available data for the interval. The average residual error is calculated for each possible interval by dividing the residual sum with the number of measurement points, and the one with the smallest average residual is chosen as the interval and trajectory to use.

A more efficient way to find the start and end points could probably be implemented, but as at most approximately 400 datapoints will be used for least squares fitting for 1400 possible time intervals, the trajectory fitting is easily run in a few milliseconds, allowing realtime implementation on a regular desktop computer. Therefore, this is not an urgent point of improvement. For comparison, a particle filter type approach was also tried, but to avoid premature convergence to the wrong hypothesis, the amount of particles needed meant that the same amount of points had to be evaluated as with this brute force approach, so that there was no net gain in performance.

The average residual error is summed over all three dimensions, so large distinct minimum jerk type motions in one or more dimensions will take precedence over noisier, less distinct motions in the remaining dimension(s). As it is the same motion, it should start and end simultaneously in all dimensions, and any deviation from this can be viewed as noise and ignored. In order to get the trajectory in position space, the polynomial is integrated,

using zero initial velocity and the last known position as the constants of integration. For an illustration of acceleration measurements and the fit function, see Figure 6.10.

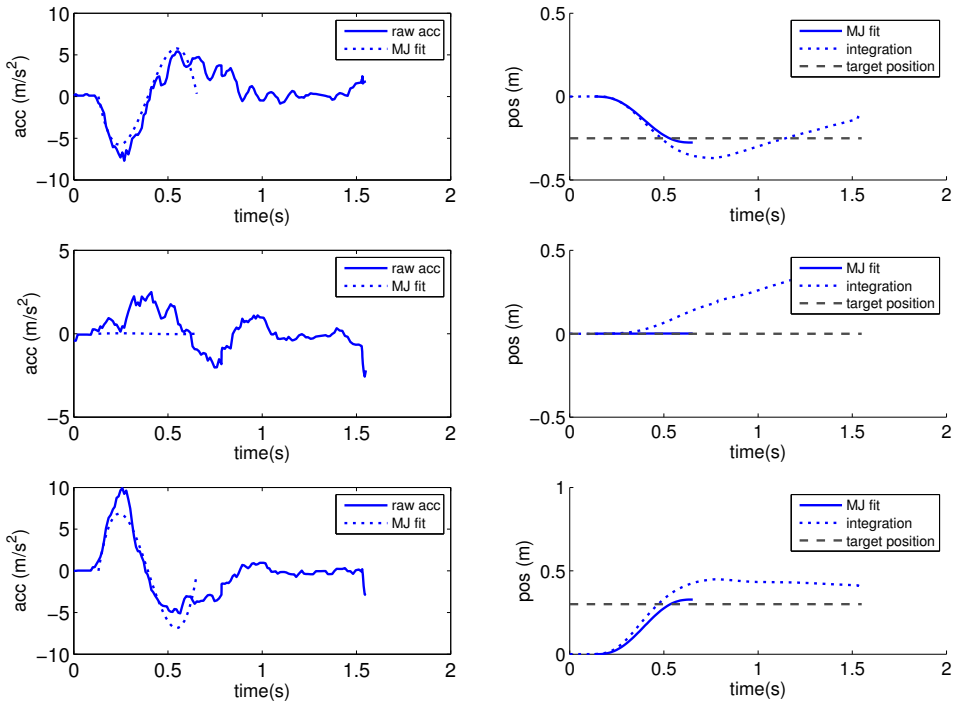


Figure 6.10: The left figure shows the third degree curve fit to measured acceleration. The right plot shows the same data integrated to position space. The dashed horizontal lines in the right figure show the positions of the target.

When the endpoint t_1 has been passed, the system will not attempt to detect the next motion until 100 ms have passed, in order to avoid treating the end of one motion as the beginning of the next. The limitations of this approach is that it is only possible to track (series of) isolated motions. It is not expected to be well suited for following complicated continuous paths, as 100 ms of motion would be cut out between each detected MJ trajectory, resulting in severe drift as the robot moves less than the user. However, here we are only interested in isolated motions, so this will not be a significant limitation for this application.

To have a ground truth for comparison, the absolute position of the wiimote was tracked with the stereo camera system, by attaching a small (5×5 mm) colored patch to the base of the wiimote. The tracking error when using the camera system like this is up to 1 cm, but drift free. The errors in the camera tracking are uncorrelated to the errors in the accelerometer tracking, and will cancel out when averages are taken over several hundred trials.

The position as tracked by the motion capture system, the position as tracked by the MJ based tracker, and all raw measurement data were logged. For comparability, each

motion was analyzed separately, rather than measuring the total error after all motions were completed.

The results of the MJ based tracker were compared to the motion capture positions, and the error, expressed as the distance between the two positions at the end of each motion when the wiimote had stopped at the target ball, was recorded. For a baseline of comparison, the raw data logged from the wiimote was parsed offline through two alternative tracking methods.

First, the result obtained when performing a naïve double integration of the acceleration was calculated. In this case, position and velocity was reset at the beginning of each motion, to attenuate buildup error. Motion start and end times manually picked from the motion capture tracker were used for limits of integration. This minimized the buildup error by excluding data not from the actual motion duration.

A more refined approach is to use a Kalman filter for the tracker. The same acceleration data as with the naïve integrating approach was used, and the filter parameters were tuned manually offline in order to minimize the error on this dataset. This guarantees that we achieve the best possible Kalman filter results, and minimize the negative impact of filter tuning choices. In an online implementation, a Kalman filter would perform worse than this.

Results

The resulting errors at the motion endpoints are summarized in Table 6.4. As can be seen, the MJ based approach outperforms the other two for precision. This is so even though the MJ based approach detects motion start and endpoints automatically. This detection has to be done manually for the other systems, to prevent small errors in acceleration to build up to considerable drift in velocity and position.

Table 6.4: Results from reaching experiment

Method	avg error	std error
Naïve integration	0.097 m	0.108 m
Kalman Filter	0.074 m	0.082 m
MJ model	0.061 m	0.051 m

Observations and Conclusions

The average errors in the previous section are for isolated motions, to be more easily compared. If we instead look at the built-up error over several consecutive motions, as illustrated in Figure 6.11, we see a much larger drift in the naive approach and the kalman filter approach than the MJ model, as the two former methods do not enforce motions ending at zero velocity, so that the velocity error eventually builds up resulting in a quadratic drift of position. The performance when simply taking the double integral of the accelerometer readings on the same data does no longer fit in the plot after the first few seconds, but the accumulated position error is 62.7 m and the velocity error is 3.18 m/s at the end of the 37 second session. The Kalman filter performs slightly better, but the errors are 14.0 m and 0.86 m/s, respectively.

The main reason for the much smaller error in the MJ approach is that it forces all motion estimates to end at zero velocity, meaning that the error in velocity is bounded, and the error in position will only grow linearly in time. In this particular scenario, adding

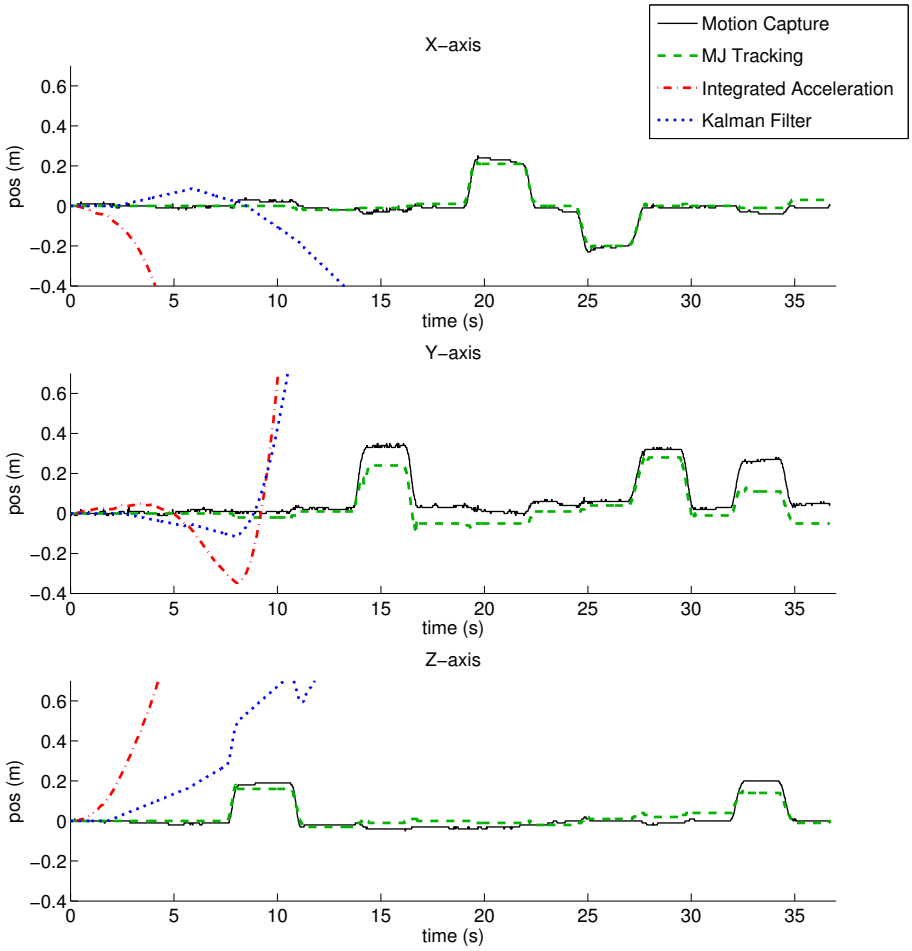


Figure 6.11: The trajectory of the wiimote estimated with the MJ approach compared to the trajectory recorded by a motion capture system. The trajectories estimated by naive double integration and kalman filtering are shown for comparison.

further boundaries, e.g. limiting positions to the robot's workspace, could probably reduce errors further.

The conclusion to draw from this study is that tracking using MJ models improves significantly over kalman filter target tracking, and that it should be possible to use the wiimote for position control of a robot arm with acceptable accuracy, at least if motion is limited to consecutive point-to-point type motions.

6.3 Teleoperated Drawing

The third offline experiment deals with what is anticipated to be the most difficult task — estimating user inputs during free continuous motion. In this experiment, we perform

initial trials with visually guided teleoperated linetracing, and do offline analysis of the input signals to determine if they can be predicted with sufficient accuracy. The purpose of this experiment is to examine if MJ models can be used to predict user input as a way to handle time delays. Thus, we want to see if we can implement a system to calculate an estimate of the user's control signal $\hat{u}(t)$, given that we know $u(t-\tau)$, where τ is the time delay.

Task Description

A prototype teleoperated drawing task was designed, and a setup was constructed so that an operator could control the robot with video feedback through arbitrary time-dealys.

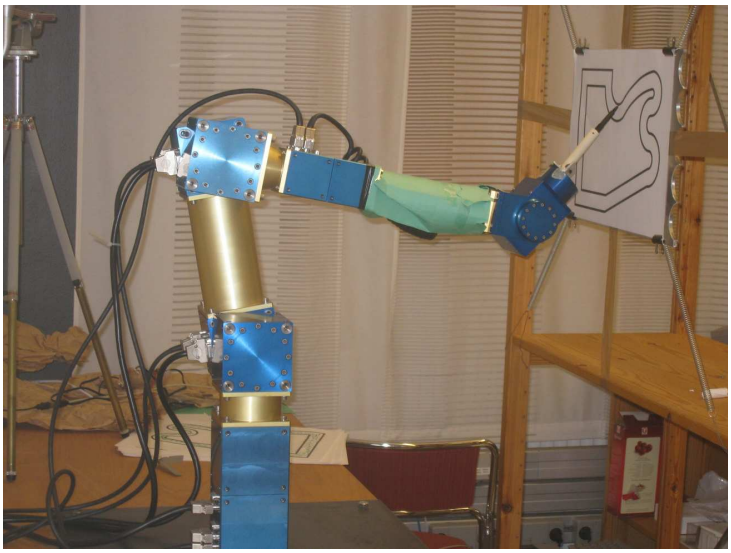


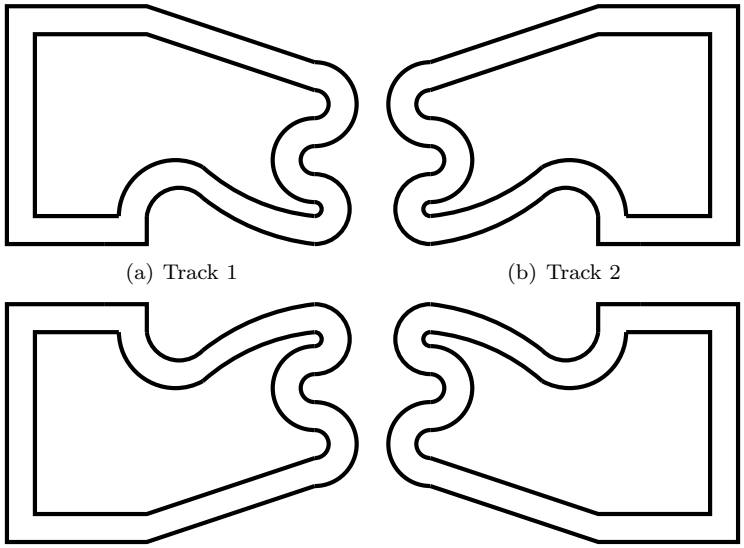
Figure 6.12: The manipulator with pen and drawing surface.

A drawing surface was mounted in the robot workspace, and the robot was fitted with an end effector that can hold a pen, see Figure 6.12. The user's task is to trace a path as fast as possible on a paper with the pen, controlling the robot with the Omega haptic interface. The path the subject was given to trace contains a mixture of straights, sharp corners, and curves of varying radius. Four different tracks were constructed by mirroring the first track about the horizontal or vertical axis, see Figure 6.13. The feedback to the user was given as unprocessed video of the end effector in the drawing surface, see Figure 5.11.

To collect initial data, the task was performed without time delays, and all inputs were recorded. Track 1 was circled 7 times, both clockwise and counter clockwise, for a duration of approximately 100 seconds.

Minimum Jerk Estimation

Processing the recorded trajectory data, Minimum Jerk (MJ) submotions were detected using a method similar to that described in Section 6.1. Since the motion itself is expected to follow a 5th degree polynomial curve, the velocity profile is expected to follow a 4th degree polynomial curve that starts and ends at zero value, with zero first derivative. Such a curve is symmetric around the apex. Empiric analysis of recorded motion data shows that



(a) Track 1 (b) Track 2

(c) Track 3 (d) Track 4

Figure 6.13: The tracks used in the tracing task.

by using a Kalman filter to smooth the velocity estimate, the point of maximum velocity $t_{v_{max}}$ is easy to detect. A threshold value requires the peak velocity to be above a certain magnitude to be registered. Using a least square approach, the 4th degree velocity profile can be fit around this peak, using only data before the peak for fitting, see Fig 6.14. The zeros of the polynomial found with this approach are used as candidates for the start and end of the motion, and used when fitting the motion data to a MJ trajectory.

A weakness in this approach is that half of the motion has to be observed before a prediction can be made. In order to facilitate earlier predictions, the Kalman Filter observer can be used to predict when $t_{v_{max}}$ will be reached. This involves estimating higher derivatives, and is prone to high uncertainties in the presence of observation noise. In practice, this means that a stable estimate of $t_{v_{max}}$ can be achieved after approximately one third of the motion, which coincides with the time when we expect to be able to make reasonably accurate predictions, as shown in Section 6.1. The implementation used here therefore uses a $t_{v_{max}}$ predicted from the EKF until $t_{v_{max}}$ is reached, after which the algorithm switches to using the observed value.

When the peak velocity has been passed, at times $t > t_{v_{max}}$, the polynomial estimate of velocity is subtracted from all subsequent incoming velocity measurements up to time t_1, and the algorithm tries to find the next velocity peak. When t_0 and t_1 are known, along with the start position $x(t_0)$ of the motion, Equation 3.2 can be fit to the latest measured data points with a least squares fitter. If we call the resulting polynomial $\hat{x}(t)$, robust trajectory prediction for time $t + \tau$ is possible by calculating $\hat{x}(t + \tau)$. For values where $(t + \tau) > t_1$, we specify that $\hat{x}(t + \tau) \equiv \hat{x}(t_1)$.

The predictions achieved this way are tracked by an extended Kalman filter, treated as observations. When there are no predictions available, i.e. at the beginning of a MJ submotion, or during motions too slow to trigger the threshold value for the MJ detector, an ordinary EKF extrapolation from the (delayed) measured position data is used as observation. See Figure 6.15

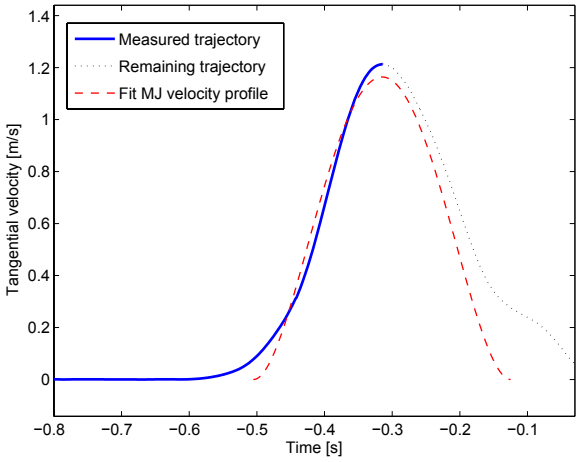


Figure 6.14: 4th degree MJ velocity profile fitted to Kalman filtered velocity data. The solid line is the data used for fitting, the dotted lined is the (unused) remainder of the measured velocity, and the dashed line represents the fit MJ velocity profile.

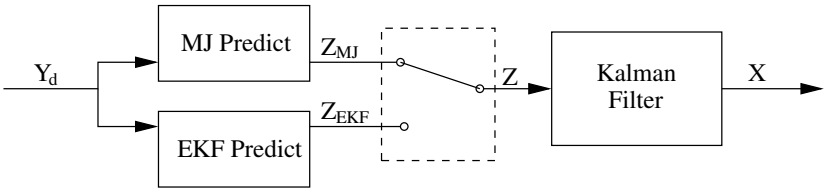


Figure 6.15: A schematic of the combined MJ-EKF predictor. The selector enclosed in the dashed box chooses signal Z_{MJ} when there is an MJ prediction available, and Z_{EKF} otherwise. The estimate X is sent to the robot controller as a setpoint.

The filter state X is defined as

$$\mathrm{X} \;=\; \{x, y, v, \theta, \dot{v}\}, \tag{6.5}$$

where x and y are the cartesian position coordinates in the plane of the paper, v is the tangential velocity, θ is the direction of motion, and $\dot{v}$ is the tangential acceleration. The motivation for this polar representation of velocity is that the MJ trajectories are defined by 4th degree polynomials in tangential velocity, making the implementation more straightforward.

In this case, we assume that the direction of motion remains unchanged for each MJ submotion, so that the observation Z_{MJ} will be the value of the MJ polynomial added to the last measured position in the direction of motion:

$$Z_{MJ}(t) = \begin{bmatrix} x(t-\tau) + (x_{MJ}(t) - x_{MJ}(t-\tau)) \cdot \cos(\theta) \\ y(t-\tau) + (x_{MJ}(t) - x_{MJ}(t-\tau)) \cdot \sin(\theta) \\ \dot{x}_{MJ}(t) \\ \ddot{x}_{MJ}(t) \end{bmatrix} \tag{6.6}$$

where $\dot{x}_{MJ}$ and $\ddot{x}_{MJ}$ are the first and second derivatives of x_{MJ} with respect to time. The observation matrix H_{MJ} is then

$$H_{MJ} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.7}$$

For cases when there are no MJ predictions available, an ordinary EKF extrapolation from the (delayed) measured position data is used as observation. Given the delayed measurements Y_d:

$$Y_d(t) = \begin{bmatrix} y_{d,1} \\ y_{d,2} \\ y_{d,3} \\ y_{d,4} \\ y_{d,5} \\ y_{d,6} \end{bmatrix} = \begin{bmatrix} x(t-\tau) + w_1 \\ y(t-\tau) + w_2 \\ \dot{x}(t-\tau) + w_3 \\ \dot{y}(t-\tau) + w_4 \\ \ddot{x}(t-\tau) + w_5 \\ \ddot{y}(t-\tau) + w_6 \end{bmatrix} \tag{6.8}$$

where w_i is measurement noise, we model the observation Z_{EKF} as:

$$Z_{EKF}(t) = \begin{bmatrix} y_{d,1} + y_{d,3}\tau + y_{d,5}\tau^2/2 \\ y_{d,2} + y_{d,4}\tau + y_{d,6}\tau^2/2 \\ \sqrt{(y_{d,3} + y_{d,5}\tau)^2 + (y_{d,4} + y_{d,6}\tau)^2} \\ tan^{-1}((y_{d,4} + y_{d,6}\tau)/(y_{d,3} + y_{d,5}\tau)) \end{bmatrix} \tag{6.9}$$

The observation matrix H_{EKF} is then:

$$H_{EKF} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{6.10}$$

Results

This proposed model was then applied offline to the measured inputs. For comparison, we also applied a predictor using the same EKF tracking but no MJ predictions, just the extrapolations from measured data. This is a near-optimal predictor without human motion models.

During the experiment, a total of 317 distinct MJ motions were detected by the system. As described above, a valid prediction is available after $\frac{1}{3}$ of the submotion has been observed. With a time delay τ, this means that it is available at time $t_0 + \frac{1}{3}t_1 + \tau$, and will remain valid until it is possible for a new submotion to dominate the trajectory, which according to empiric study of operator input was determined to be approximately 200 ms after the peak of the current MJ submotion. Observed submotions on this setup are typically 200–400 ms in length, so the upper limit for τ where we will still have meaningful prediction will be expected to be around 100 ms, if we assume the lower limit of 200 ms for motions.

Setting τ to 100 ms, we observe that the MJ prediction errors are smaller than the errors caused by the delay, or the errors we get by applying the EKF predictor based purely on input measurements, See Table 6.5. The errors were measured at points were a valid MJ prediction was available, and the error measurement E_i for each predictor function P_i was defined as follows:

Table 6.5: Comparison of mean square errors of the MJ model and pure EKF predictions.

	Delayed signal	Non-MJ EKF	MJ Predictor
Mean square error E_i	145.63 mm^2	$13.609mm^2$	$8.372mm^2$

$$E_i \equiv \frac{1}{n}\sum_{k=1}^{n} \|\mathrm{X}(t_k) - P_i(t_k)\|^2 \tag{6.11}$$

Where $\mathrm{X}(t)$ is the true input signal at time t, with position given in millimeters, and t_k, $k = \{1 \ldots n\}$ are all time points where a valid MJ prediction exists. In order to quantify the error that was caused by the delayed signal without a predictor, a predictor function defined as the position at the time $t - \tau$ was used:

$$P_{delay}(t) \equiv \mathrm{X}(t - \tau) \tag{6.12}$$

where τ is the time delay, in this case 100 ms.

The difference in the magnitude of the error between the pure EKF approach and the MJ is not very large, but there is a difference in the quality of the error. Figure 6.16 shows both predictions as compared to the real signal and the delayed signal of the x coordinate. With the MJ curves there is only a small amount of oscillations as the input slows down after

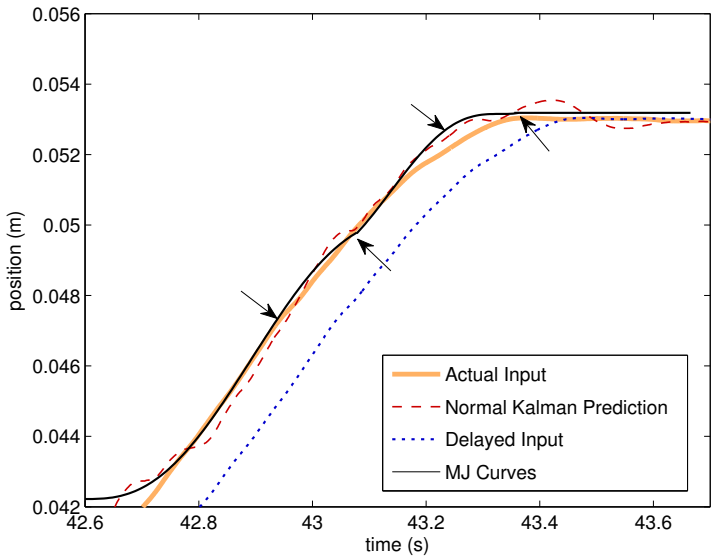


Figure 6.16: A comparison between MJ curves and pure EKF prediction. The MJ trajectory in the plot contains two submotions. The arrows pointing from the left show where each submotion would become available to an online prediction system, the arrows pointing from the right show where each prediction would lose validity, as it would be possible for a new submotion to dominate the trajectory.

a period of higher velocity. This is where one of the strengths of the MJ-based prediction is shown as it enforces an MJ trajectory, which by construction does not oscillate. In this particular setup, this is an important property, as oscillations at higher frequencies may be transformed into low-frequency oscillations with magnitudes several times higher by the manipulator controller.

Note that while Figure 6.16 shows two entire superimposed MJ submotions, in online usage, due to the time dely, only the later part of each submotion would be available as a prediction. In the figure, the arrows pointing from the left show where each prediction would become available, and the arrows from the right show where the prediction would lose validity as a new submotion could possible become dominant. Since the MJ trajectory is also tracked with a Kalman filter, the influence of the MJ prediction remains for some time after this point.

An example of the final result of the compound prediction, using MJ predictions when these are available, and simple EKF type extrapolation from the delayed signal when no MJ predictions are available is shown in Figure 6.17. This plot shows the x coordinate as Track 4 is traced one lap clockwise.

Observations and Conclusions

The results from the predictions made by applying the MJ model to the offline data are promising. Time delays up to 100 ms can potentially be bridged with this predictive approach. Longer delays are mostly untractable, and a theoretical upper limit that will never be breached with this approach is of course the total length of a MJ submotion, or 200 ms. These figures correspond to roundtrip delay times on trans-atlantic internet connections.

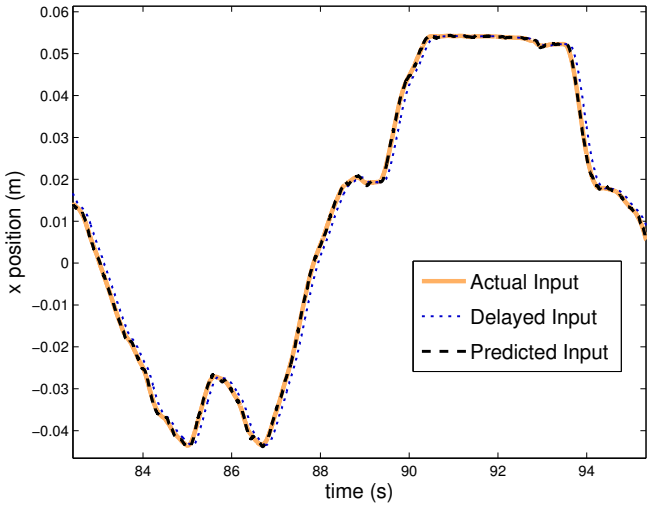
Another observation to be made is that a regular EKF without any human motion models can also be used for meaningful prediction of user input at these timescales, at the cost of more high-frequency noise, which may or may not constitute a problem, depending on the remainder of the controlled system. However, one should not expect a straight-forward EKF implementation to perform well at predicting over any longer time delays.

6.4 Conclusions

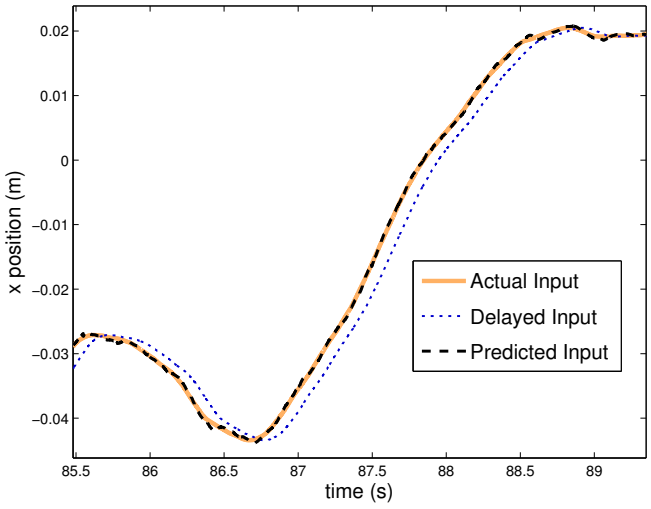
As shown in the three offline experiments in this chapter, it is possible with simple means to use minimum jerk models to estimate the input from a human operator. Offline, with the freedom to tune parameters to fit recorded data, these estimates can be used to make reasonably accurate predictions for time delays of up to approximately 100 ms.

The predictor performance depends heavily on how well the user input fits the model. In these experiments, this corresponds to whether the input can be accurately described by one single MJ submotion, or if several submotions are needed. This is especially noticable in the comparison between the ballcatching experiment and the wiimote experiment. In the former, the reaction time is short, and many subjects make an erroneous first estimate of where to catch the ball, and one or several subsequent corrections are necessary. In the wiimote reaching case, the target is stationary and visible to the subject for a long time before the reaching motion is initiated. Thus, the subject has no difficulty in making a correct first estimate, and no corrections are needed. These observations are in accordance with the theory, as described in Section 3.2.

For the third experiment, with continuous free motion input by the operator, the theoretical support for applying the MJ model is weaker than with the other two experiments, as the motion is constrained to two dimensions, and there is no single target to reach for.



(a) Predictor performance with 100 ms delay.



(b) Predictor performance with 100 ms delay, closeup.

Figure 6.17: Example performance of prediction.

The results show that accurate predictions can be made using the model, but the there is no conclusive support for the performance being more accurate than a model-free Kalman filter approach. However, the predictions made by the MJ model produces a smoother path, which may be preferable for some applications.

As predictors, the methods implemented here may seem unecessarily crude. However, it should be noted that for linear processes, such as tracking motion in cartesian space, a Kalman filter (which is an optimal estimator) is equivalent to minimizing the square of the

error [149], which in turn is equivalent to performing a least square fit to the measured data.

All in all, the results of all three offline experiments are promising, and the following chapter will test these approaches in online implementations.

Chapter 7

Online Experiments

In this chapter, we present the results of *online* experiments. By this, we mean that estimation and prediction is done in real time, and that the estimated or predicted values are used directly in the control of the robot system. Performing experiments with human subjects on a physical setup and not only in simulation is important as a proof of concept — showing that the models and techniques can be applied in reality, without simplifying assumptions about signal quality or subject behavior. Online implementations also enforce strict causality for all filters and estimators. To show generality of results we always try to repeat experiments over several subjects.

7.1 Ballcatching Experiment I

This experiment has multiple objectives. First, we want to examine the task of teleoperated ball catching, which to our knowledge has not been previously attempted. We want to acquire a baseline of typical performance figures for the task, in order to assess its difficulty, and have a basis for comparison for different user aids or time delay compensation systems using human motion models. We also want to see if the task difficulty and user performance can be altered with the help of user aids, and to assess the performance of an online implementation of a minimum jerk based estimator for user input.

Initial Prototype Study

To get an initial indication of the difficulty of the problem, a trial where a subject in Pisa, Italy controlled the robot in Stockholm, Sweden was performed using a portable version of the 3D monitor interface installed on a laptop computer and the Omega haptic device. Otherwise, the interface setup was identical to the one described as setup 1 in Section 6.1.

In this experiment, due to time limits, only a few tries could be made, and of these only a single ball was successfully caught — probably the first ever instance of successful transcontinental teleoperated ballcatching. However, the difficulty was evident. Even with a robot arm as fast as the one we had at our disposal, it could not match all the fast motions the user could make with the haptic interface. The input from the device was scaled by a factor of 7.7 before being relayed to the robot, and the robot is not 7.7 times faster than a human subject. This led to ideas concerning different types of aid systems that could be applied in order to simplify the task.

Predictive User Aids

As shown in Section 6.1, the robot is fast enough to catch a thrown ball when it gets an early, correctly predicted intercept point from the stereo camera tracking system. If it can be given an early prediction of where the user aims to move, it could move there in a smooth trajectory in equal or less time than it takes for the user to move to the same point after executing a number of corrective submotions.

Therefore, we propose two different kinds of predictive user aids to be tried in the experiments, alongside a straightforward implementation where the user has unaided direct control of the robot. The control systems used in the experiment are:

- **System A - Direct Control**

  In the direct control setup, position and velocity data from user input is transmitted directly as setpoints to the robot controller, at 50 Hz. The controller then accurately mimics the user's movements up to the dynamical limits of the robot. This should provide us with a baseline for the performance of teleoperated catching.

- **System B - Minimum Jerk Prediction Based Control**

  The minimum jerk (MJ) prediction based control system fits MJ trajectories to the user input data whenever a velocity threshold is exceeded. This was empirically set to 0.07 m/s. The MJ model is then used to predict the goal in time and space of the current user motion. A prediction is made when a velocity peak has been detected, leaving the second half of the motion to be predicted using data from the first half. The predicted goal of the user motion is sent as a setpoint for the robot arm instead of the current motion. The prediction is done with least square fitting, as described for the offline experiment in Section 6.1.

- **System C - Semi-Autonomous Control**

  In this setup, the predicted ball trajectory from the stereo camera tracker is also given to the controller. The system is based on system B, and works identically, until a predicted user input is found to be within 10 cm of the expected ball trajectory, when the user is assumed to attempt to catch the ball, and an autonomous ball-catching system is engaged. The autonomous system sets the robot controller setpoint to the point on the expected ball trajectory that is the closest to the present position.

Experiment 1 - Online Prediction

The robot was fitted with a cylindrical end effector for catching balls, see Figure 7.1. The diameter of the cylinder is 14 cm, and the balls are soft juggling balls with a diameter of 6.6 cm. To ensure repeatability of the experiment, and to ensure that the difficulty is unchanged between subjects and setups, the balls were launched using a mechanical launcher with a precision of ± 10 cm for this distance (see Fig 7.2). The balls were sent according to a pregenerated pattern that was the same for all trials.

For this experiment, we use the user interface designated as setup 1 in Section 6.1. Each subject was first shown the simulator version of the teleoperation system to allow him or her to get acquainted with the user interface. The relevant features of the graphics display were explained. These include the ball flying toward the robot, and the projected trajectory of the ball as in earlier experiments, but also a home position that the end-effector must be brought to before each throw. The latter was introduced to ensure that all subjects started in the same position before attempting to catch a ball. The system was designed

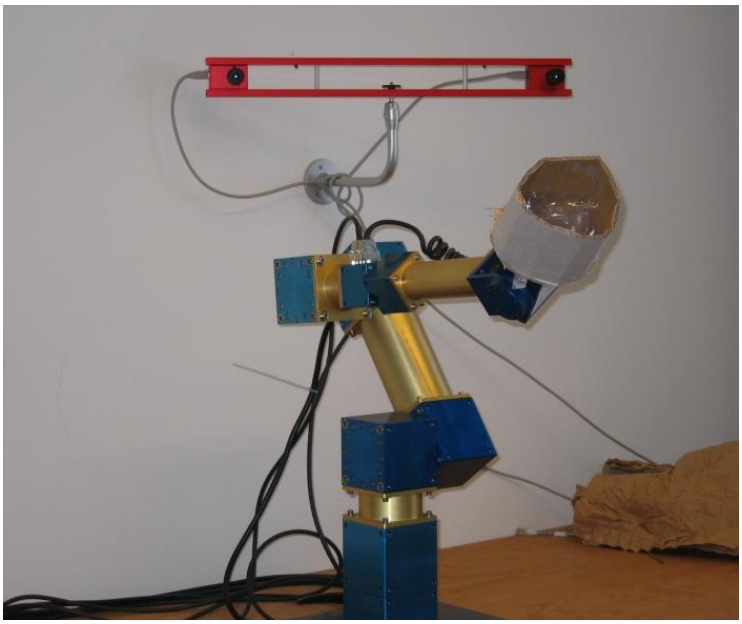


Figure 7.1: The robot arm with ball-catching end effector.

so that ball catching could not commence before the robot had been brought to the home position. Successful homing was indicated by a short sound signal. After the introduction, the subject was allowed 20 practice throws with simulated robot dynamics and prerecorded balls, as in the offline experiment.

Before engaging the robot hardware, further instructions regarding the operation of the physical robot were given. The subject was also informed that the audiovisual feedback of the user interface would be the same for catches as for rim hits, but that the result of each throw, catch, rim or miss, would be called out by one of the experiment leaders.

The sequence of throws with the different tested control systems was explained to the subject, along with the question to be answered after trying out each system, and the fact that there would be more detailed questions after the experiment. The subjects were not informed of the underlying control model for each of the systems. In this way, we hoped to be able see whether there was a significant difference between the systems, and also if unbiased subjects would prefer a system that catches more balls or a system that gives more control to the user.

The sequence repeated for each control system was as follows:

- Twelve unrecorded practice throws followed by a pause while the balls were collected.
- Three times twelve recorded throws with a pause after each series of twelve.
- The question "How well did you perceive that you could control the robot with this system?" with possible answers being an integer score ranging from zero (worst) to five (best).

The order in which the systems were presented was permuted between subjects as to not introduce a bias from learning curve effects. After the last control system was tested, the

Figure 7.2: The mechanical ball launcher

subjects were asked a set of more detailed questions to which there were no fixed ranges of allowed answers:

- "What system was the easiest to use?"
- "What did you perceive to be the difference between the systems?"
- "What was good or bad with each version?"
- "General comments. What improvements would you like to see?"

All user input via the Omega device, ball trajectory data from the tracking system, and actual robot position were logged. Also, all experiments were recorded with a video camera, so that all subject actions and comments as well as actual ball and robot behaviors could be studied post-experiment.

Also noted were the subject's age, sex, and familiarity with ball sports, video games, and radio controlled models. A total of 25 subjects were used. These had no prior experience of robot control. 12 were female, 13 male, and the ages ranged from 19 to 63 years. All the subjects were persons unaffiliated with our lab that had responded to an advertisement.

Experiment 1 - Results

The first performance measure considered is the performance of the online MJ predictor. The overall performance is summarized in Table 7.1. The first MJ trajectory that has been found is evaluated. We define a measure we call *improvement factor*, which shows how much a prediction has improved the knowledge of the final position. We define the improvement

Table 7.1: The performance of MJ prediction in experiment 1.

Quantity	average	std
Movement Duration	0.249 s	0.0895 s
Improvement Factor	17.2%	82.3%

Improvement Factor	Trials within ratio
>25%	44.9%
>50%	26.0%
>75%	5.9%

factor IF as the portion of the remaining distance that is correctly bridged by a prediction, as in Equation 7.1:

$$IF = \frac{|x(t_p) - x(t_1)| - |\hat{x}(t_1) - x(t_1)|}{|x(t_p) - x(t_1)|} \tag{7.1}$$

where t_p is the time when the prediction is made, $\hat{x}(t_1)$ is the predicted endpoint of the motion, and $x(t_1)$ is the actual endpoint of the motion. The improvement factor thus shows how much closer the predicted position at t_1 is to the later measured position at t_1, compared to the remaining distance when the prediction is made. Thus an improvement factor of 0 would mean no improvement, and 100% would mean that the system had made a prediction that was exactly the same as the actual value. Note that predictions in the wrong direction give negative improvement factors. The number of tries for which the predictor improves the position by at least 25, 50, and 75% are also given.

Comparing the catching performance of system A, where the MJ predictions were not used to control the robot, but merely logged, with the MJ predictor performance, we find that good MJ prediction correlates to successful catching, with $p = 0.010$. If we only examine the successful catches from system A, we find that the average improvement of MJ predictions is 32.2%. There was also a significant correlation between the reaction time and the performance of the MJ predictor, with $p = 0.0124$. The faster the reaction, the better the MJ prediction.

The second performance measure considered is the number of balls that were caught with the different systems, as shown in Table 7.2. The figures here are the average over all tries in each experiment, i.e. 900 tries for each system in experiment 1. The unexpected result in experiment 1 is that subjects performed worse with the semi-automatic system C than with the other systems.

For reference, the robot system was also run in fully autonomous mode without any user interaction, for 108 balls in the same pattern as with the human subjects. In this case, the reaction time is on the order of 100 ms, and the initial error that needs to be corrected is minimal. The catching performance was that 73% of the balls were caught, 19% bounced off the rim of the end effector and the remaining 8% were missed completely. This is the hardware limit on performance for the set of ball trajectories used in the experiments.

Defining the performance as the ratio of balls actually caught is an aggregate measure of the performance of human and machine. In order to discriminate between the subject's and the system's performance, each subject's logged input signals were examined. There are two major components of the quality of user input, the spatial precision of the input, and the reaction time, good input being both fast and accurate.

Since the spatial accuracy depends on the initial distance from the manipulator and thus the user's hand position to the ball trajectory [39], we formulate the accuracy A as in Equation 7.2:

Table 7.2: The percentage of balls caught in experiment 1.

Trial	Balls Caught
System A	20.6%
System B	16.4%
System C	15.7%

$$A = \frac{d_c}{d_i} \tag{7.2}$$

Where d_c is the distance between the ball trajectory and the setpoint for end effector when the ball was the closest to the manipulator, and d_i is the initial distance between the setpoint of the end effector and the trajectory, before the motion begins. In Figure 7.3 we see that the subjects produce more accurate input with system A than with system C.

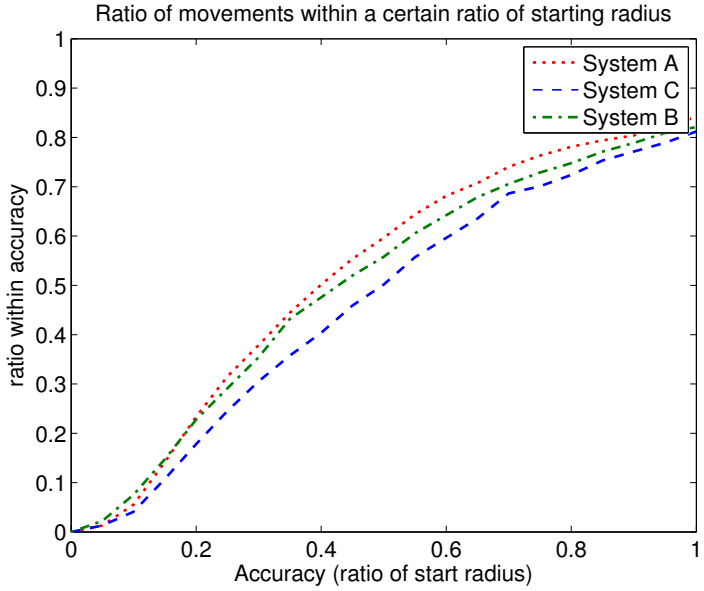


Figure 7.3: The distribution of accuracies for the different systems in Experiment 1. The plot shows the accumulated ratio of attempts that were better than a certain accuracy.

With system C, the earlier a catch attempt was recognized and the autonomous system was engaged, the higher the probability for catching. This relationship is illustrated in Figure 7.4. The plot shows an almost linear relationship between the distance left for the ball to travel and the probability for the ball to be caught.

One way to measure reaction time that is relevant to the current setup is to see when a prediction within 10 cm of the final intercept point was made with system C. Figure 7.5 shows the distribution of remaining distances to the interception point when such a prediction was made, and the autonomous mode engaged.

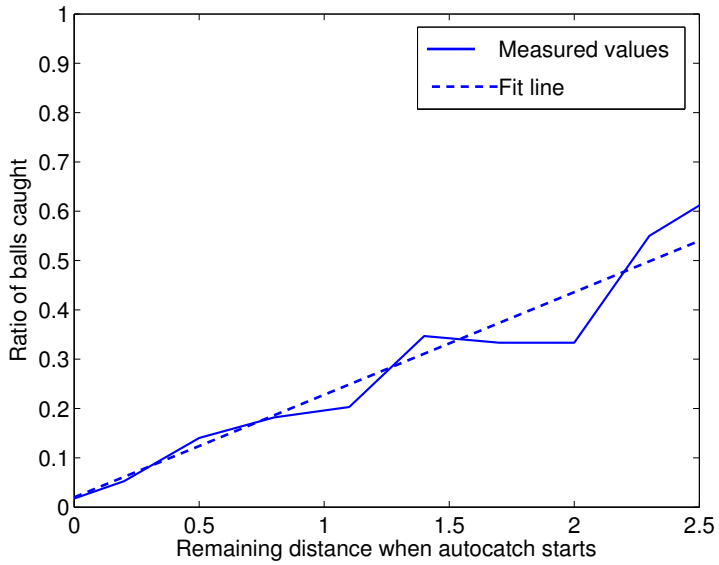


Figure 7.4: The probability that a ball was caught as a function of the remaining distance (from the ball to the interception point along the ball trajectory) when the autonomous system was engaged. The dashed line shows a linear fit to the data.

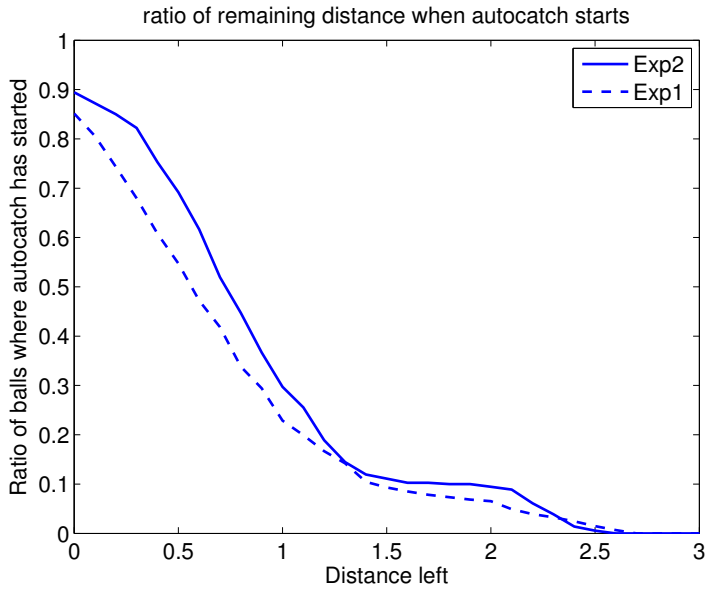


Figure 7.5: The distribution of the ratio of trials where a catch attempt has been detected at different distances to the interception point.

Table 7.3: The performance of MJ prediction in experiment 2.

Quantity	average	std
Movement Duration	0.248 s	0.0875 s
Improvement Factor	19.6%	71.3%

Improvement Factor	Trials within ratio
>25%	46.0%
>50%	23.9%
>75%	5.8%

Experiment 2 - User Instruction

The results from Experiment 1 were not as expected, as the most direct control system, A, was more effective than the others in terms of the number of caught balls. We found that the raw user input was worse when using system C than when using system A, indicating that, at least when the subject was uninformed about them, autonomous catching motions degraded the subject's performance, as the subject tries to compensate for perceived discrepancies between user input and robot motion.

To investigate whether the same effect would be present even for subjects who know what to expect from each control system, *Experiment 2* was performed with ten new subjects who had been given a brief explanation of the different systems. Because time constraints forced a smaller number of subjects, the number of possible permutations of the order presentation of three systems was too large. Therefore system B, which is neither the most faithful to user commands, nor potentially most effective, was eliminated from Experiment 2.

Compared to Experiment 1, the modifications in Experiment 2 were:

- 10 subjects, only systems A, and C.
- Subjects were informed about the differences between the systems. They were told that system A would give them direct control of the robot, and that system C would make autonomous corrections if necessary to catch the ball.
- In system C, the motions needed for autonomous catching was aborted after the ball had passed, eliminating irrelevant differences between user input and robot motion. In the previous version, the robot would keep moving towards the intercept point until it was reached, regardless of if the ball was still possible to catch or not.
- Also in system C, autonomous catching was indicated in the user interface whenever it subsumed user input, by changing the color of the end effector to green. This was expected to make it easier to understand what was happening.
- The only question asked was a general request for comments after the completion of the tests.

For experiment 2, the performance of the online MJ predictor is summarized in Table 7.3. The first MJ trajectory that has been found is evaluated. The average duration is given, and also the average improvement, as explained in the previous section. These figures do not differ much from experiment 1, except the average relative improvement which is slightly better, with a slightly smaller standard deviation.

As in experiment 1, there was a significant correlation between the reaction time and the performance of the MJ predictor, with $p = 0.0029$, but good predictions and successful

Table 7.4: The percentage of balls caught in experiment 2.

Trial	Balls Caught
System A	18.0%
System C	24.2%

catching in system A only correlates with $p = 0.14$, with an average improvement factor of 30.0% for successful catches.

The second performance measure considered is the number of balls that were caught with the different systems, as shown in Table 7.4. The figures here are the average over all tries in each experiment, i.e. 360 tries for each system in experiment 2. The noteworthy result is seen in comparison with experiment 1: the number of caught balls varies with the information given to the subject. For system C, there is an increase in the number of balls caught when the subject is told that (s)he will be assisted by the system, while for system A, there is a smaller decrease in performance when the subjects are told that they will not be assisted.

In experiment 1, we find that the subjects' inputs had noticeably lower accuracy for system C than for the other two systems. Therefore, the poor performance of system C is assumed to be due to this poor performance in the user input. The distributions of accuracies for the different systems in experiments 1 and 2 are shown in Figure 7.6. Several subjects in experiment 1 complained that system C was difficult to understand and did not perform as they expected, and tried to overcompensate when they perceived the robot to move erroneoulsy. Examining the experimental logs, we found several cases of a subject moving the robot to miss the ball by about 9 cm, with the semi-autonomous system being activated too late to catch the ball, only moving the robot a few centimeters closer to the ball trajectory before the ball had passed. The user would then complain that they would have caught the ball if the robot had not made "that strange jerky motion just before the ball passed".

Also, as can be seen in Figures 7.6 and 7.5, the user input performance is better for system C in experiment 2. Regarding reaction speed, since the velocity is close to constant, this figure can also be expressed in terms of time. For experiment 1, the median time left when a catch attempt was detected was 185 ms, in experiment 2, the median remaining time was 225 ms, which is 22% longer.

Observations

Apart from the aforementioned results, there are some further observations of interest that were made in the experiments that help to put the results in context.

In experiment 1, the users were asked to rate the different systems by answering the question "how well did you perceive that you could control the robot with this system?", on a scale from 1 to 5. System A and B scored an average of 3.3, while C was given a 2.6 on average. They were also asked to state which system was the easiest to use. Here, 12 subjects stated system A, 8 preferred system B, and 3 preferred system C. Two subjects failed to notice any difference between the systems. The relationship between preferred system and the performance of the subject on the different systems can be seen in Table 7.5. The correlation between preferred system and best performing system is noticeably weak. Specifically, we note that out of the 6 subjects who performed best with system C in terms

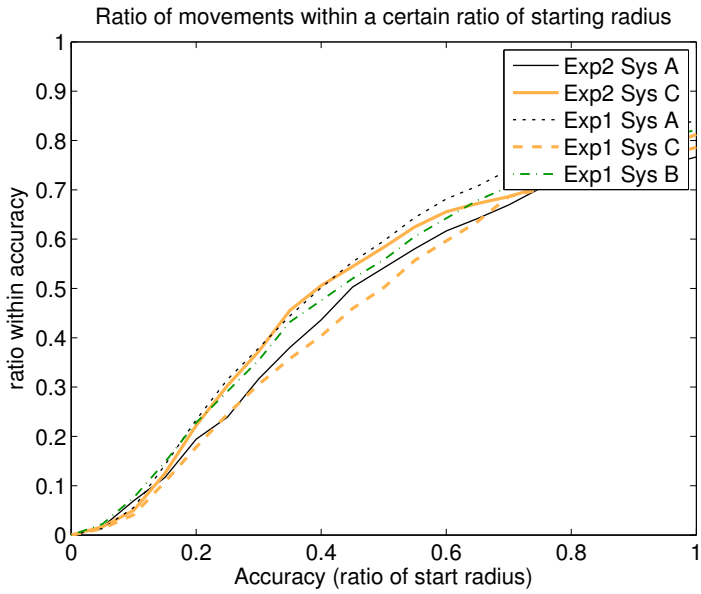


Figure 7.6: The distribution of accuracies for the different trials. The plot shows the accumulated ratio of attempts that were better than a certain accuracy.

Table 7.5: The relationship between preferred system and performance

system	Subjects who preferred system	Subjects who performed best with A	B	C
A	12	8	1	3
B	8	2	4	2
C	3	1	1	1

of the number of balls successfully caught, only one subject actually preferred system C. We assume that the dislike of system C contributed to the subjects' poor performance with this system.

When given a chance to describe their preferences freely, many subjects stated that they felt that system C reacted slower, was less precise and had a tendency to exaggerate movements. A was in most cases perceived as faster and more precise. When told how they had performed with the different systems after the experiments, the 3 subjects who had performed better with C but preferred A expressed surprise at this, as they were convinced that they had caught more balls with system A.

Also, many subjects were frustrated by their overall poor performance, and their enthusiasm for the experimental task decreased visibly after a sequence of several missed attempts.

Data concerning age, sex, and experience of computer games, ball sports and, remote controlled models was collected from the subjects. Of these, age was the only one found to have a significant ($p<0.05$) correlation to ball catching performance, with younger subjects performing better.

7.2 Ballcatching Experiment II

The objective of this experiment is to see if minimum jerk (MJ) based estimation models can enhance the performance of time-delayed teleoperation of a dynamic task (ball catching).

In the previous experiment, we used input prediction to generate straight paths towards the intercept point in order to allow the robot to have enough time to move there. A drawback of this approach was that the motions that needed the aid the most, i.e. the ones that contained large subsequent corrections, were also the most difficult to predict, resulting in little or no performance gain.

In this experiment, we instead focus on a scenario where we assume that a prediction will be more relevant, teleoperation under time delay.

Control Structure

For comparison, we implement a classical delay compensation system and compare this to our approach using model based prediction of user input.

As detailed in Chapter 2, a classical approach to delay compensation for teleoperation systems is prediction of the feedback signal, commonly using Smith prediction. For our implementation, which we refer to as system A, we apply both a Smith predictor and Kalman filtering (to predict the ball trajectory) as in [105]. With the Smith predictor approach, the remote slave site is simulated locally at the master site, and measurements are used to correct the simulation to prevent it from diverging from the actual state of the remote site. In the employed scheme, everything displayed to the operator through the user interface is generated interacting exclusively with the dynamic simulation running on the operator interface computer. Since this interaction does not involve a time delay, the risk of instabilities is significantly reduced. This limits the approach to tasks where all significant aspects of the environment can be modeled with sufficient accuracy.

To close the control loop over the robot via the communication link, we use a non-linear, multivariate Smith predictor control structure (depicted in Figure 7.7). Robot motion is predicted by feeding the user input into a simulated model of the robot that also uses the same code as the real robot controller, allowing very accurate prediction. The ball trajectory is predicted using the same Kalman filter as is implemented in the ball tracking system.

The command and measurement communication delays, τ_c and τ_m respectively, are handled in two parts. The first part handles the stochastic aspect of the delay by adding an artificial delay τ_a to the command delay τ_c when a command packet arrives at the robot so that their sum, the virtual delay τ_v, is constant, as described in [84, 21]. In effect, the stochastic delay is exchanged for a slightly longer deterministic delay. The internet time delay is stochastic, with a high variation. The communication packets are transmitted at 100 Hz, or at 10 ms intervals, so it is expected that a small portion of packets will be delayed more than τ_v, or even long enough to arrive after the succeeding packet, c.f. Section 5.4. These packets are treated as dropped and ignored. Since the average time delay may vary slowly over time, τ_v is continuously recalculated. The allowed rate of change is very slow compared to other time constants in the system, typically τ_v can change with a millisecond or less over several hundred packets.

The simulation result $y_{sim}(t + \tau_v)$ is then delayed by $\tau_v + \tau_m$ before it is compared to the measured state $y(t - \tau_m)$ to form the simulation correction δ. This means that when a measurement packet arrives from the robot, the current value of τ_m is calculated from timestamps and an old simulation result $y_{sim}(t-\tau_m)$ retrieved from memory for comparison.

The net effect of this compensation is that the simulation, the haptic controller, and the operator all perceive time τ_v ahead of real time. That allows the generated command

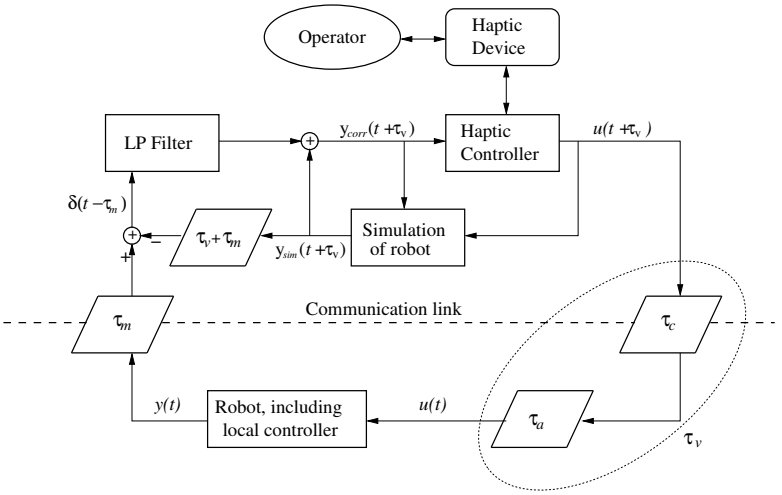


Figure 7.7: Control structure A, with Smith predictor only (skewed boxes represent delays).

signal u to travel to the robot before it is needed by the robot controller. Values of τ_v upto 200 ms have been tested successfully on the setup without instability.

The correction signal δ is low pass filtered to reduce zero order hold noise that would otherwise be felt by the operator as vibrations of the same frequency as the communication link, 100 Hz, given that the local controller in the UI runs at 2.2 kHz.

The largest drawback with this approach is the need to simulate the entire roundtrip delay, including the artificial padding. The time difference between the simulation and the measurements is thereby $\tau_v + \tau_m$. This does not pose a large problem for aspects of the remote site that are easy to simulate, such as the robot dynamics or ball in ballistic flight. However, it is problematic when unexpected events occur. For instance, there is no advance warning for when a new ball is launched, and the trajectory can therefore not be simulated until the ball has already been in flight for a time corresponding to $\tau_v + \tau_m$. Thus, the operator's reactions will be delayed by the entire roundtrip delay.

We therefore also propose a slightly different control structure, which we call system B. Instead of handling the entire roundtrip delay when predicting the remote state with a simulation, the delay handling is split into two separate predictors, as suggested in Section 3.1. The simulation of the remote site needs only bridge the shorter delay τ_m, while the delay in the outgoing signal τ_v is handled by predicting the future control signal, $\hat{u}(t + \tau_v)$. The principal structure for this approach is shown in Figure 7.8. With this approach, the operator's reaction to an unexpected event will only be delayed by the one-way delay τ_m. The design of the input predictor is described in the following section.

Experimental Setup

The setup was modified slightly as compared to the previous experiment. Where not explicitely stated to differ, the setup was identical to the previous experiment. First, the motion controller of the robot arm was modified to become faster. This was possible by limiting the robot's allowed workspace to a smaller area (50×50×40 cm) centered on the homing position, and then optimizing the controller for the dynamics of this limited region. In the 4560 attempts (including practice sessions) recorded in the previous experiment, no balls

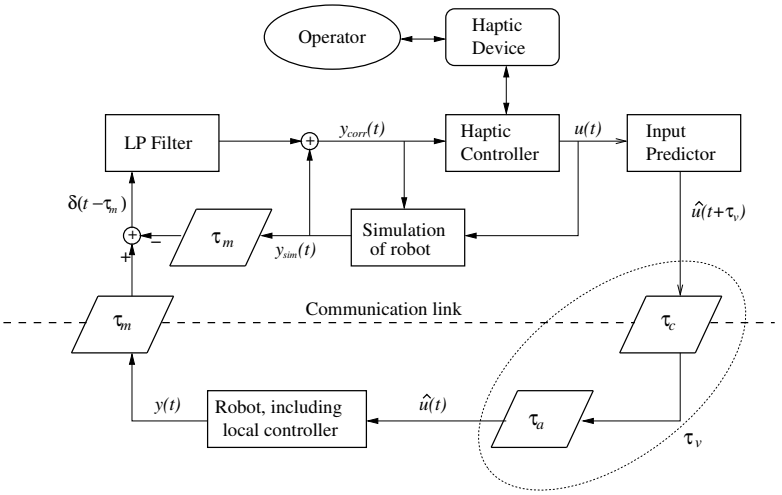


Figure 7.8: Control structure B, with Smith predictor and input predictor (skewed boxes represent delays)

had been successfully caught outside of this area, so this limit should not have any effect on ballcatching performance. This optimization allowed the robot to accelerate up to 50% faster in the limited workspace making it easier for the robot to keep up with the input from the human operator.

Third, after noting that balls that hit more than 4 cm from the center of the ballcatching cylindrical end effector had a tendency to bounce off the rim and not be caught by the robot, we added a small corrective function to the robot controller to ensure more successful catches. By using the predicted ball trajectory, we had the robot automatically center on the trajectory if it was brought within 5 cm of it. This effectually increases the radius of the target to hit for guaranteed catches from 4 to 5 cm, and the effective target area is increased by 56%. This was effective at all times, independently of other user aids or delay compensation.

The performance in the presence of time delays was tested using actual internet time delays, for maximum realism. For practical reasons, both the slave and master sides were located in the same laboratory, but all communication signals between the two were delayed according to actual internet delays (including completely dropped packets) sampled during 24 hours starting from 11:40 GMT, Sep 25 2008. Since we have seen in Section 6.1 that the upper limit for successful prediction of user input using a skilled subject is approximately 70 ms, we use internet connections with average delays no greater than this magnitude.

One sample set consisted of delays measured between the Royal Institute of Technology (KTH) in Stockholm, Sweden and the University of Genua (UGE) in Italy. The median one-way delay time for these packets was 26.6 ms, and the standard deviation was 59.1 ms. The other sample set was generated by timing packet travel times from KTH to Georgia Institute of Technology (GT), in Atlanta, Georgia, USA. These packets had a median travel time of 63.0 ms, but with a much lower standard deviation of 0.2 ms. A deterministic one-way time delay of approximately 2.5 ms was added by the rest of the communication loop.

Table 7.6: Results from ball catching experiment

Setup	**Success rate ($\pm 2\sigma$)**
No delay	0.288 ($\pm$0.045)
System A, UGE delay	0.256 ($\pm$0.062)
System A, GT delay	0.160 ($\pm$0.052)
System B, UGE delay	0.275 ($\pm$0.063)
System B, GT delay	0.280 ($\pm$0.065)

Experimental Procedure

10 Novice subjects were asked to catch thrown balls using the teleoperation setup. The subjects were student volunteers, all male, ages from 23 to 30. They had no prior experience with the setup, and were not told any details of the control systems prior to the experiments. The subjects were given a brief instruction to the user interface and the ball catching task, and were allowed to get acquainted with the setup by practicing catching balls until they felt comfortable with the task. This took between 30 and 50 tries for most subjects.

In the experiment, for each subject 5 different settings were tried, with 20 throws per setting. The ordering of the settings were randomly permuted between subjects. The settings were:

- No delay, used for comparison.
- System A, UGE delays
- System A, GT delays
- System B, UGE delays
- System B, GT delays

To ensure repeatability of the experiment, the balls were launched using the same mechanical launcher as in the previous experiment. The balls were sent according to a pregenerated random pattern that was the same for all trials.

Results

The performance results are presented in Table 7.6. With the longer GT delay times, the performance with System A drops to approximately half the success rate that was achieved with zero delays, while the success rate of system B drops by less than 0.01 compared to the zero delay baseline. This result is statistically significant at $p < 0.05$.

With the shorter UGE delays, the success rate of System A drops by 0.03, while the success rate of System B drops by 0.01 as compared to the zero delay baseline. However, here the difference in performance is not large enough to be statistically significant.

The average virtual delay τ_v used by the system was 31.8 ms for the University of Genua delays, and 69.3 ms for the Georgia Tech delays. In practice, this meant that 20% of the University of Genua packets were dropped, c.f. Figure 5.13(a), and less than 0.1% of the Georgia Tech packets. Given the inertia and reaction times of the robot, and the frequency content of the human control signal (which is typically less than 10 Hz), dropping 20% of all packets does not have a significant effect on performance.

Observations and Conclusions

These results show that the MJ predictor approach can successfully bridge time delays of more than 60 ms, while a traditional Smith predictor approach encounters problems. A probable cause for this is that, with the Smith predictor, transient events such as the firing of a ball with the ball launcher will be delayed the entire roundtrip time, or approximately 130 ms in the case of GT delays. Given that the subject only has a few hundred milliseconds to react, this delay is too large for successful task completion. On the other hand, with the MJ approach, the transients will only be delayed by half as much, giving the user just enough time to react. As shown previously, user input can be predicted successfully up to 70 ms, so this is close to the limit of what the MJ predictor approach would be expected to cope with. In geographic terms, the difference of the two approaches is if a dynamic internet-based teleoperation task is limited to the same continent (within Europe), or if it is feasable to perform it intercontinentally (between Europe and North America).

7.3 Wiimote Control Experiment

In this experiment we test if a robot manipulator can be controlled in position space using a wiimote game controller if we apply a minimum jerk (MJ) model to the input. The goal is to find out how the MJ based input tracker performs when applied online, and to examine how well an untrained subject can use this potentially intuitive input device.

Based on the results from the offline analysis in Section 6.2, we assume that applying naive integration of the acceleration measures, or even Kalman filtering of the same, will not make safe control of the robot possible, and therefore we only try MJ based approaches.

Experimental Setup

The subjects used the robot to carry out the same target-touching tasks that were done directly with the wiimote in the offline experiment described in Section 6.2. The colored ball targets described in the same Section were placed so that they were reachable by the robot. The target balls were mounted on PVC pipes so that they were situated at 25 cm intervals 30 cm above the robots centered default position. The targets will move when hit, but come att rest in the original position within a few seconds. See Figure 7.9 for an illustration of the setup.

The end effector used in this experiment is a floorball mounted on a 10 cm piece of PVC pipe. Since the targets are contained within the limited 50×50×40 cm space used for the experiment in Section 7.2, the same optimized, faster motion controller was used.

Aid systems

In the offline experiment we had ideal conditions for the MJ model assumptions with the subjects physically touching a clearly visible target. In the present experiment, the subjects need to perform a correct mapping of their hand motion to the robot workspace. We assume that errors in this mapping may potentially decrease the precision. Since the diameter of the floorballs is 6 cm, the center of the end effector needs to come within 6 cm of the center of a target in order to hit it, and the average precision in the offline experiment was ± 6.1 cm. Therefore, two types of aid systems were also implemented and evaluated. These systems grant the user less freedom of motion, but give higher precision for the task. The result of using unconstrained MJ tracking will be compared to the results with the different aid systems.

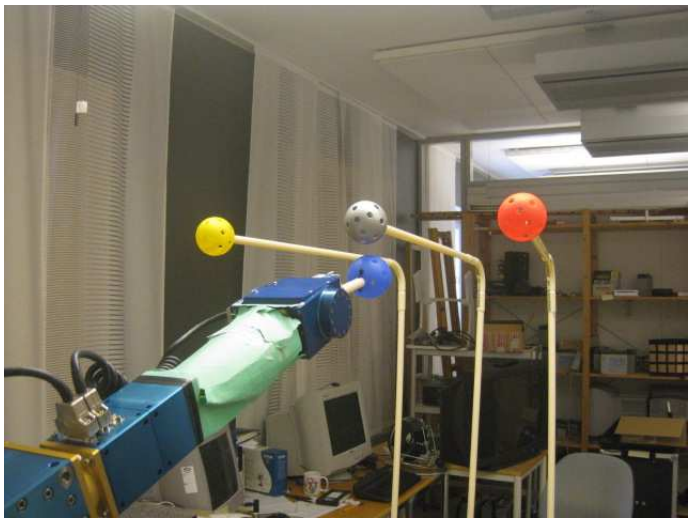


Figure 7.9: The manipulator and targets. From left to right, the target colors are yellow, gray, and red.

The first aid system used virtual fixtures. The resulting motion from the MJ based system is projected onto a straight line from the starting point to each of the possible goal points. The line with the shortest euclidian distance to the estimated position is chosen. Using this approach, the possible motion trajectories of the robot arm are along a straight line towards one of the three goal points, and the user can control which goal to move towards, when and how fast to do this, and how far towards the goal to move.

The second aid system implemented task control, for the highest possible precision. In this control mode, the user's input is used to choose between 4 possible predefined tasks:

1 Touch red target

2 Touch gray target

3 Touch yellow target

4 Stay in start position

In order to choose which task to execute, a polynomial trajectory estimate is calculated as in the MJ control system, and the endpoint of the motion is extrapolated. The target closest to the endpoint is then touched. If the endpoint of the motion is not close enough (within 15 cm) to any of the targets, task 4 is chosen and the arm remains motionless. In the first few centimeters of a motion, before an accurate target estimate is available, the arm will follow the directly tracked position of the wiimote. The arm will therefore start moving almost at the same time as the user, long before the user's motion is completed, and if the user moves straight towards one of the possible target postions, the result will be an exact mimic of the user's motion in real time. If the extrapolated endpoint is recalculated to be closer to another target during the execution of a motion, the task will be changed in mid-motion.

Table 7.7: The percentage of hits with each of the three systems.

	MJ Control	MJ + Virtual Fixtures	MJ + Task Control
median	37.5%	72.5%	85%
best	75%	97.5%	100%
worst	17.5%	37.5%	42.5%

Experimental Procedure

15 subjects were used for this experiment, 12 male and 3 female. All subjects were right-handed and had normal or corrected to normal vision. They were given a brief instruction in how the control system worked, and were asked to hold the wiimote so that its position paralleled the position of the robot. Before the experiment started, they were given a few minutes free practice to become acquainted with controlling the robot.

Each subject was then asked to touch the colored balls according to an order called out by the experiment leader. This was the same order as used in the offline experiment. 40 ball-touching motions were carried out for each control system, and the control systems were tested in the same order for each subject, beginning with the unaided MJ controller, and continuing with the virtual fixtures and ending with the task based controller. Since we have observed in the ballcatching experiment that user instruction may have a large impact on performance, all subjects were given a brief instruction telling them that the first system would follow their motions directly, the second would use virtual fixtures to limit the possible directions of motion and the third would be task-based, autonomously touching the ball that the user had been identified as trying to touch.

Results

The wall-mounted motion caption system we employ is only usable within the robot workspace, and it was therefore not possible to use this for ground truth measurements when users were controlling the robot. We assume, however, that the precision of the MJ based tracker in this experiment is similar to what was shown for the experiments in Section 6.2. If we isolate the three best performing users, we see that their average errors — as measured as the distance from the actual target position — are only slightly larger than the inherent error in the tracker. Their average error was 11 cm.

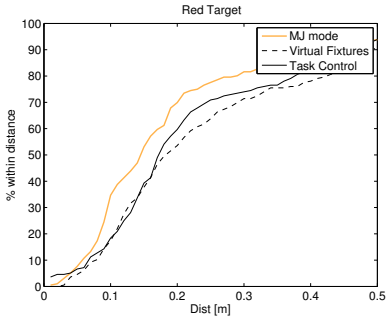
The average error over all 15 subjects was 19 cm in total, but the error in the horizontal plane was smaller than the error in the vertical direction, 11 cm as opposed to 16 cm. A probable reason for this is that moving too far in the vertical direction would still score a "hit", while a miss in a horizontal direction resulted in a "miss", and the subjects would take more care to correct this. We assume that the error mostly comes from the subjects finding it difficult to map the wiimote position space to the robot position space with high precision, as they receive no haptical feedback.

The task completion performance was measured in a "hit-or-miss" fashion. That is, each time the user would hit the indicated target in one try without hitting any other targets counted as a hit, and a failure to do this, however close, counted as a miss. This means that the allowed margin of error is the diameter of the balls, 6 cm. The results are given in Table 7.7.

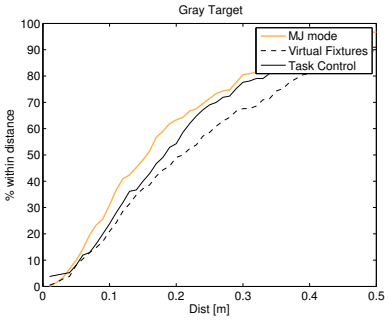
While the "hit-or-miss" metric gives an overview of the performance, analyzing the tracked positions of the wiimote controller gives quantitative measure of precision. Figure 7.10 shows the cumulative percentage of tries that came closer than given distance of

the targets. The most eye-catching result is that the performance is significantly worse for the yellow target, which had the right-handed subjects reaching to their left, which seems to be a less natural motion for many subjects. However, the lack of motion capture data for this experiment makes it difficult to state of this caused the subjects to reach with less precision, or if the resulting motion was less accurately described by the MJ model.

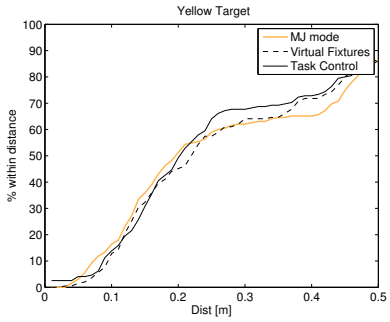
Another observation is that for the most part, the user input has higher precision for the MJ mode. This is to be expected, as the subjects are likely to notice that they can complete the tasks with less effort when different levels of aids are added.



(a) Cumulative performance for red target.



(b) Cumulative performance for gray target.



(c) Cumulative performance for yellow target.

Figure 7.10: User performance

Some subjects were fairly close to the targets when they missed in MJ control mode. The performance of these subjects improved greatly from the added aid of virtual fixtures or task control, the largest improvement being from 22.5% success without aids to 87.5% and 92.5% success rates for the two aided modes.

Other subjects, with a larger standard deviation in their input, did not improve their performance with added aids. Indeed, some subjects' performance even degraded. They made motions that were too large and rapid, in effect saturating the acceleration sensors. They thus miss the targets by more than 15 cm. The effect of this is that the robot does not move as they expect it too. They make the false assumption that the motion was too small for the control system to register, and adjust by making even more exaggerated motions, further worsening the performance.

After a post-experimental brief reminder to use motions of the same magnitude as they want the robot to use, these subjects were able to perform as well as the rest. However, that

they did not manage this on their own is an indication that the lack of intuitive feedback when in task mode is a likely cause of this behavior.

Conclusions

When estimating hand trajectories from the accelerometer data gathered from a wiimote, imposing minimum jerk motion models improves performance significantly over simpler approaches, enabling position control. However, the precision is still not good enough for any but the simplest of manipulation tasks. Adding virtual fixtures greatly improves precision while retaining enough sense of direct control to allow users to improve their input motions. Even the completely novice users in the experiment could achieve good success rates after a few minutes of operation.

Giving more control to the robot — as in task control mode — raises the success rates further for most users. However, this control mode is less intuitive and users that have poor performance with this mode do not understand how to improve.

Given these results, it is reasonable to expect future applications utilizing these types of intuitive controllers for manipulation tasks. Several simple modifications could also be added to the approach described here in order to raise precision. The major drawback of accelerometers is the inherent position drift. This problem can be greatly reduced by adding sensors that give absolute position measures, even if these are of very low signal quality.

As suggested for other applications, a tetrahedronal IR array can be used to give absolute 6 DOF pose measurements when the wiimote is centered on the array [87]. Using sensor fusion, this could be combined with the approach described here to cover a large range of possible poses. Also, it is highly probable that the newly announced wii MotionPlus, a gyroscope sensor extension for the wiimote, can be used to further improve motion tracking, as it would provide information on angular velocity. If we assume other input devices than the wiimote, many cell phones, PDAs, and other portable devices that have accelerometers are also equipped with cameras, that can be used to attain simple measures of motion or position.

7.4 Drawing Experiment

In this experiment we test the effects of time delays and minimum jerk (MJ) based prediction on teleoperated drawing, using the same setup as in Section 6.3. Here, we predict the entire roundtrip delay time using user input prediction, instead of splitting the prediction in half between input and feedback as in the experiment described in Section 7.2. The fundamental difference between these approaches is illustrated in Figure 3.1.

Motivation

The motivation for using only input prediction is that there are cases when feedback prediction may be difficult to obtain, perhaps due to difficulties modelling the remote environment. A typical case is when the only feedback we have of the remote site is a video signal from a camera that is arbitrarily placed. If the remote camera is mounted in an unknown position, which may not even be stationary, it can be very difficult to accurately model the robot in the image, as needs to be done in systems for feedback prediction.

Experimental Procedure

Experiments were carried out where subjects were asked to trace the path clockwise. First 10 laps for practice, then 10 laps that were recorded. This was repeated 3 times, with a different setting each time. Possible settings were:

A No communication delays added, no predictions. This is the control case that the other settings are compared to.

B 100 ms roundtrip time delay added, no predictions. This case is used to determine the effects of delays on the completion time of this particular task.

C 100 ms roundtrip time delay added, input prediction system active. This case is used to determine how well the predictions can cancel the negative effects of the time delay.

The order in which the three settings were presented to the subjects was permuted between subjects to cancel out overall learning effects. The subjects were told that "three different settings" would be evaluated, but not what those settings were. They were asked to trace the track as fast as they could without making mistakes. A mistake is defined here as drawing outside the lines that define the track. A total of 12 subjects were used. The subjects were all male, ages 23–31, and did not have prior experience with the experimental setup.

We measure "Objective telepresence", as defined in [5]. This means that the task completion time is used as the performance measure. Of the measured tries, the median time for successful trials was recorded. The median is chosen to eliminate outliers. Results for subjects who did not successfully complete at least 5 laps with each setting was not considered. Success is defined as never drawing outside the borders of the track for the duration of the lap.

It should be noted that even in setting A, with no added communication delay, there are some inherent delays in the system. The time needed to process input data and calculate predictions is less than one millisecond, but the robot processes commands at 100 Hz, giving the "current" commanded setpoint a delay that varies from 0 to 10 ms. Similarly, the video display is updated at 15 Hz, so that the image being displayed will have a varying delay of 0–67 ms, plus the overhead for image acquisition and processing. None of these delays were treated in any special way, but were added to the total delay of the system for all three settings.

Results

For all subjects but one (subject 10), the median completion time was lower for setting C, with the predictor, than for setting B. One subject even performed times that where better than the control setting A, See Figure 7.11. This is possibly the effect of some overall learning, with subjects performing relatively better on the last setting they were presented with. A small overall trend for improvement over time was observed. When corrected for setting differences, the subjects were on average 2% faster on the second trial, and 6% faster on the third. The average completion time was 12.86 s for setup A, 19.3% longer for setup C and 40.0% longer for setup B. We examine the statistical significance of the results by performing a Student's T-test on the results, and find that the performance with setup A is significantly better than those of both setup B (with $p = 1.83 \cdot 10^{-12}$) and setup C (with $p = 3.54 \cdot 10^{-6}$). The performance with setup C was significantly better than that of setup B (with $p = 0.000187$).

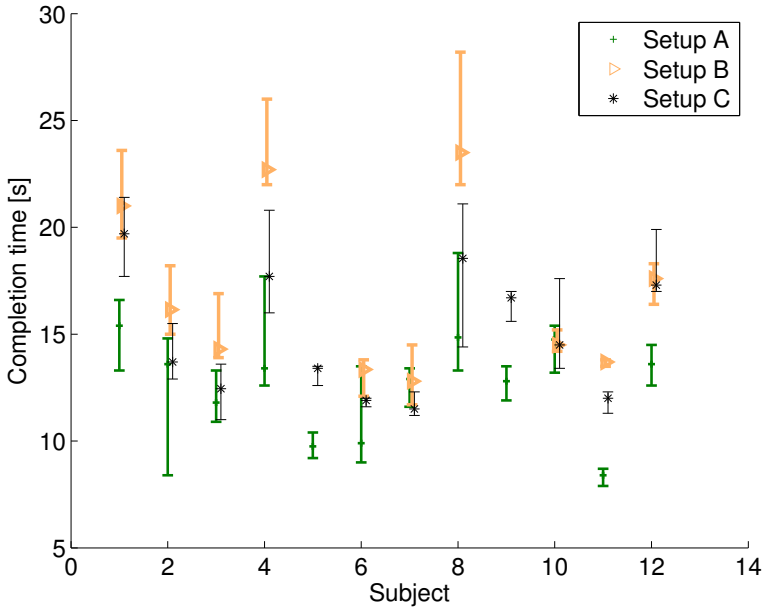


Figure 7.11: Task completion times for the 12 subjects. The plot shows median completion times and 90% confidence intervals. Note that subjects 5 and 9 did not complete the task for setup B, and are not included when calculating statistics.

Two subjects, 5 and 9, were only able to complete one successful lap with setting B, and are therefore excluded from these results. All others were able to complete at least 5 successful laps with each setting. Comments from several subjects stated that the main difficulties they perceived with settings B and C were oscillations, most subjects did not explicitly notice that there was a time delay.

Observations

One strength of using the relatively simple MJ model for predicting human input is that since we do not use any knowledge of the particular task being performed, it should generalize to most possible vision guided tasks.

However, there are several limitations to this approach. The first is that since new MJ submotions can be generated as often as every 100 ms, it is not realistic to make predictions much further into the future than this, as there is a real possibility that the operator will be moving along a trajectory that we have not yet detected. A 100 ms roundtrip delay is comparable to the internet roundtrip delay between Stockholm, Sweden and New York City, USA. However, when using the Internet, time delays can behave stochastically, something that would have to be considered and dealt with in an implementation over an internet connection, possibly using an approach similar to that in Section 7.2.

In the presence of longer delays, but where we have a plausible model for predicting the feedback from the remote environment, it is possible that the input predictor model described here could be combined with a feedback predictor as in Section 7.2. Then, one could successfully predict both input and measurements, and thus bridge larger time delays

than when just predicting one of the two.

Another limitation is that the MJ model assumes free motion. Its assumptions do not hold under the presence of outer forces, e.g. contact forces. Therefore, this approach is not applicable without modification for tasks that require haptic interaction with the remote environment. However, as shown in Section 6.3, even without the MJ predictions, a pure EKF predictor has similar magnitude of tracking errors. It remains as future work to examine how this type of approach can be extended to tasks that include contact. It is possible that hybrid approaches, utilizing parts of this approach and parts of traditional delay-handling approaches may be the solution. One could conceive of using simple remote autonomy to avoid unwanted collisions. If an operator's typical reaction to contact forces is known, it could be possible to incorporate this in a predictor at the remote site.

Finally, there this is yet no treatment of stability issues for the MJ prediction approach. Although empirical trials have not shown any unstable behavior, a more rigorous analysis remains to be done.

Conclusions

In this section, we have described a method that uses predictions of user input to cancel the negative effects of time-delays on teleoperation. The experimental results show that this is a valid approach, as the performance of all but one of the 10 subjects who completed the task improved as compared to the control case with the same delay but no predictive system. The main strength of this approach is that we only need a model of the master side of the telerobotic system, while the slave side can be left completely unmodeled. This means that once we have designed the master system, we can deploy a slave system in novel environments to perform novel tasks without the need for recalibration or redesign of the control system. Furthermore, once the predictive system is calibrated for one master system, it could be used with any slave system that works in the same coordinates. This also allows for easy-to-deploy remote sensors, such as arbitrarily positioned uncalibrated cameras.

In the implementation presented here, the prediction system relies on MJ models, but any model that sufficiently well describes human motion could conceivably be used.

7.5 Conclusions

These applied experiments have shown successful implementations of MJ model based estimators. Hereby, we have proven the viability of such approaches. By using several different subjects for each experiment, we have shown the generality of the models.

It should be noted that the experiments presented here represents proof of concept, and not final applications. For the input estimation models used here to be truly viable as functioning systems, a complete systems integration aspect should be taken, adding error detection and fallback when the model-based estimator does not track correctly. Also, there is a need for thorough user studies, where aspects of user learning and interface accustomization are also taken into account, as results are likely to differ between novices and experienced users.

Chapter 8

Conclusions

In this chapter we provide a short summary of the thesis, present the general conclusions, and suggest some interesting problems that remain for future investigation.

8.1 Summary

This thesis treats the subject of using human motion models to create estimators for the input signals of human operators controlling a telerobotic system.

Problem

In telerobotic systems, the control signal input by the operator is often treated as a known quantity. However, there are instances where this is not the case. For example, a well-studied problem is teleoperation under time delay, where the robot at the remote site does not have access to current operator input due to time delays in the communication channel. Another is where the sensors in the input device have low accuracy. Both these cases are studied in this thesis.

Approach

A solution to these types of problems is to apply an estimator to the input signal. There exist several models that describe human hand motion, and these can be used to create a model-based estimator. In the present work, we propose the use of the minimum jerk (MJ) model. This choice of model is based mainly on the simplicity of the MJ model, which can be described as a fifth degree polynomial in the cartesian space of the position of the subject's hand. This allows us to use simple input devices that only track the position of the operator's hand, as opposed to measuring, modelling, and tracking the entire arm configuration. Also, as the polynomial formulation is very fast to calculate, real-time implementation is simplified.

Offline Experiments

An initial series of experiments is conducted where subjects perform different tasks, and the data from the input devices are recorded. This recorded data is then used to evaluate different implementations of MJ-based estimators.

In the first experiment, we let users try to catch balls with a robot arm in a simulated environment. We compare the performance of a virtual reality (VR) setup, where the

subjects are allowed free motion, to the performance of a simpler setup consisting of a 3D screen and joystick. The VR setup was expected to comply with the required assumptions made by the MJ model.

The results show that the subjects perform similarly with both setups, and that MJ models can be used to make a more accurate description of the subject's input than simpler polynomial curves. We also see that the MJ model is a more accurate description of user input when the subject's initial attempt at catching is successful and does not not need major subsequent corrections. Allowing a subject to train with the system in order to increase the initial accuracy, we see that the MJ model fits well enough to allow accurate predictions of input trajectories on average 70 ms into the future.

In the second experiment, we let subjects touch physical targets with a wiimote video game controller. We implement an MJ-based tracking algorithm by fitting MJ polynomials to observed accelerometer readings using least squares. We see that this approach gives higher accuracy and much less position drift than using simple double integration or Kalman filter based target tracking methods.

In the third experiment we let subjects use a pen mounted on a robot arm to trace a predetermined track on a piece of paper. We apply an extended kalman filter (EKF) to the recorded input data, and compare the accuracy of predicting future input when including an MJ-based predictor in the EKF framework. We see that although the MJ-based predictor only gives a marginal improvement to the EKF in terms of mean square error, the oscillations in the error are decreased, resulting in smoother predictions. Using this approach, predictions up to 100 ms are achievable.

Online Experiments

In the final series of experiments, we use MJ-based estimators to generate control signals to a robot manipulator, evaluating the performance in real-time scenarios, and showing proof of concept. Each experiment in this section is related to one of the three experiments in the previous section.

In the first experiment, we use an MJ-based predictor to generate control signals to the robot arm, which the subjects use to catch real balls thrown across the laboratory room using a mechanical ballistic launcher. Here, we compare the performance when using the unfiltered user input to control the robot, to the performance when using the predicted endpoint of the user motion. We also evaluate an approach where let the robot perform autonomous catching when the predicted input is close enough to the predicted trajectory of the ball. As an unexpected result of this experiment, we see that subjects react poorly to the two kinds of predictive control and generate lower quality input as a result. When instructing the subjects on the different types of control systems used, they perform better when the predicted signal is used. However, we see that overall, using the predictions in this way is not very helpful, as the predictor performed the worst for the cases when it is needed the most, i.e. when the subject's initial attempt is far from catching the ball, and the subject needs to correct the motion one or more times.

In the second experiment, we tune the robot controller for faster motions, allowing the robot to catch up with the subjects' motion even when given a late start, and perform teleoperated catching with delayed communication signals. We measure the percentage of successfully caught balls and see that when the round-trip communication delays reach 120 ms, using an MJ-based predictor to predict the operator's input will outperform a more conventional controller that only uses prediction on the observations.

In the third experiment, we let subjects use the wiimote to control the robot to touch physical targets. A real-time implementation of the MJ-based estimator is used to filter the

signal. We also evaluate further constraints on the signal by using virtual fixtures and task control. We see that novice users can use this interface to perform successful reaching, and that the extra constraints improve performance. However, we also observe that placing too large constraints on the operator's input reduces the intuitiveness of the interface, and a few subjects have difficulties controlling the robot in semi-autonomous task control mode without detailed instructions.

In the last experiment, we use an input predictor based on an extended kalman filter (EKF) and the MJ model to bridge time delays as subjects perform a teleoperated line tracing task. We let the subjects perform the task both without and with a 100 ms time delay. When we include the time delay, we let the subjects perform the task both with and without input prediction. We measure the time needed to perform error-free task completion, and find that subjects need 37% longer times to complete the task with the delay, but that completion times with the input predictor present are only 19% longer than for the undelayed case.

8.2 Conclusions

There are several conclusions to be drawn from the experiments. The main, overall conclusion is that MJ motion models can be used to construct estimators for the input from a human operator, when the input is made in position space, with the operator performing visually guided reaching motions.

If we make a more detailed analysis, we find the following major results from the experiments.

- It is possible to use a human motion estimator to construct a predictor for future input. This extends the existing toolbox of available methods for dealing with time delays of different types, by adding a new possible place in the control scheme were prediction can be performed. This opens up teleoperation for scenarios where it is impossible or too resource consuming to predict the observations from the remote site.
- The MJ model is a valid model to use for such an input predictor for several different types of tasks and operator motions.
- An MJ estimator can be used to track position using only low-quality acceleration measurements, enabling position space input with devices where this would otherwise be impossible.

We also find the following minor results:

- The MJ model is a valid description of the hand motions of an operator controlling a robot with a haptic interface if there are no feedback forces. If the user does not make an initial error in reaching for a target, the model fits well using only a single submotion.
- When we introduce a predictor to enhance the performance of a teleoperator, it is important to inform the operator of the details of the system. Otherwise, the operator may be frustrated by not understanding why the robot seems to disobey direct commands, resulting in poorer performance.
- The limits of how far into the future it is possible to predict operator input using MJ models depends on the allowable error, but the error is relatively small up to 70 ms, where there is a significant increase, making this the practical upper limit for most cases.

The last point can also be generalized to all conceivable approaches toward predicting operator input. Even using other approaches, we would not expect that it is possible to make accurate predictions significantly longer than with the MJ models, as long as the operator is free to change his/her mind and switch trajectories or targets at will.

Perhaps the most significant limitation of using MJ models is that they are not valid in the presence of interaction forces. Thus, the methods described here can not be used when the operator needs to be presented with force feedback via a haptic interface. This means that the methods may not be applicable to tasks where the operator directly manipulates objects and/or experiences contact forces. Even teleoperation tasks requiring object interaction with tangible forces may contain components of free motion, for example when the subject is reaching for an object to manipulate, or moving an object from one place to another. It is conceivable that the MJ models could be applied to these types of free motion parts of interaction, while the parts that include contact forces are handled differently. This may result in worse dynamic performance while performing the parts of the task that require physical contact, but we would expect these parts to require higher precision, and therefore necessarily be carried out att lower velocities, where time delays are more neglectable.

Another limitation of the MJ-based approaches is that it is inherently difficult to accurately detect superimposed submotions when we have noisy signals. Thus, a tracker relying on only accelerometer readings is not able to track any possible human motion, but requires that the operator makes a complete stop before moving towards a new target. It is possible to practice this type of motion and learn to perform position control using this method, but since it is different from natural unhindered motion, the intuitiveness of the interface may be lost, limiting the field of possible applications.

8.3 Open Questions and Future Work

The methods described in the present thesis are only implemented and tested at a prototype level. Therefore, there remains several possibilities to study more advanced implementations, including implementations of more advanced estimation algorithms. An interesting and important study that remains is to examine how these methods work when applied to tasks taken from the real world, as integrated parts of complete systems, preferably compared to state-of-the art solutions for the same problems.

As stated above, it is not likely that the length of predictions can be extended greatly using only primitive motion models, as the free will of the operator implies that his/her intentions and objectives may change at any time during operation, and this is not likely to be predictable, at least not with pure motion analysis. Other work has been done on identifying user intent on a higher level, and it is possible that these approaches, combined with understanding of the remote environment and the possible tasks, can bridge longer timespans, at the expense of giving the user direct control. An interesting question for further study is how we can combine the relatively low-level estimators based motion models presented in this thesis with higher-level models of the task context. This could be used both by switching estimators to handle scenarios where we have varying time delays or periods of contingent loss of communication packets, as well as designing a combined estimator that uses high level task models to extend the prediction time of low-level motion models.

Another limitation mentioned above, that MJ does not handle interaction forces, can possibly be approached by extending the human motion models. However, it is unlikely that a purely kinematic description in cartesian space, such as the MJ model can be extended with interaction forces. A more likely candidate for future study would be minimum torque

change models, where interaction forces could be mapped to joint torques using knowledge of the kinematic structure and configuration of the ooperator's limbs, along with an accurate description of the dynamics involved. In many cases this would require much more tuning to and knowledge of the individual operator, but we can imagine scenarios, like space teleoperation, robot prosthetics use, or telesurgery, where the possible gains could motivate the extra resources needed.

The present work only examines one single model of human motion, the MJ model, but many of the ideas presented here regarding constrol structures could still be valid with other operator input estimators. An open question that remains to examine, is how other — either simpler or more complex — estimators inserted into the control structure would compare to the MJ-based estimators.

Bibliography

[1] Daniel Aarno. Autonomous path planning and real-time control - a solution to the narrow passage problem for path planners and evaluation of real-time linux derivatives for use in robotic control. Master's thesis, Department of Numerical Analysis and Computer Science (NADA), KTH, Sweden, 2004. TRITA-NA-E04006.

[2] Bruce A. Aikenhead, Robert G. Daniell, and Frederick M. Davis. Canadarm and the space shuttle. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, 1(2):126–132, 1983.

[3] R. Andersson. Understanding and applying a robot ping-pong player's expertise. In *Proceedings of IEEE International Conference on Robotics and Automation, ICRA*, pages 1284–1289, Scottsdale, AZ, 1989.

[4] Russell Andersson. Dynamic sensing in a ping-pong playing robot. *IEEE Transactions on Robotics and Automation*, 5(6):728–739, December 1989.

[5] Rafael Aracil, Martin Buss, Salvador Cobos, Manuel Ferre, Sandra Hirche, Martin Kuschel, and Angelika Peer. *Advances in Telerobotics*, volume 31 of *STAR*, chapter The Human Role in Telerobotics, pages 11–24. Springer-Verlag, 2007.

[6] Paolo Arcara and Claudio Melchiorri. Control schemes for teleoperation with time delay: A comparative study. *Robotics and Autonomous Systems*, 38:49–64, 2002.

[7] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, Feb 2002.

[8] K. J. Åström. Optimal control of markov decision processes with incomplete state estimation. *Journal of Mathematical Analysys and Applications*, 10, 1965.

[9] K. J. Åström and B. Wittenmark. *Adaptive Control*. Addison-Wesley, Boston, MA, 2nd edition, 1995.

[10] M. Baker, R. Casey, B. Keyes, and H.A. Yanco. Improved interfaces for human-robot interaction in urban search and rescue. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2960–2965, Oct. 2004.

[11] Billur Barshan and Hugh F. Durrant-Whyte. An inertial navigation system for a mobile robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2243–2248, Jul 1993.

[12] Antal K. Bejczy, Paolo Fiorini, Won Soo Kim, and Paul Schenker. Toward integrated operator interface for advanced teleoperation under time-delay. In *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 563–570, Sep 1994.

[13] Antal K. Bejczy, Won S. Kim, and Steven C. Venema. The phantom robot: Predictive displays for teleoperation with time delay. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 546–551, May 1990.

[14] Igor Belousov, Sviatoslav Chebukov, and Victor Sazonov. Web-based teleoperation of the robot interacting with fast moving objects. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 673–678, 2005.

[15] Jeffrey J. Biesiadecki, P. Chris Leger, and Mark W. Maimone. Tradeoffs between directed and autonomous driving on the mars exploration rovers. *The International Journal of Robotics Research*, 26(1):91–104, 2007.

[16] Peter Birchall. *The Longest Walk – the World of Bomb Disposal*. Sterling Pub Co Inc, 1998.

[17] E. Bizzi, N. Accornero, W. Chapple, and N. Hogan. Posture control and trajectory formation during arm movement. *Journal of Neuroscience*, 4(11):2738–2744, 1984.

[18] M. J. Black, E. Bienenstock, J. P. Donoghue, M. Serruya, W. Wu, and Y. Gao. Connecting brains with machines: The neural control of 2D cursor movement. In *Proceedings of the 1st International IEEE/EMBS Conference on Neural Engineering*, pages 580–583, Capri, Italy, March 2003.

[19] Theodore T. Blackmon and Lawrence W. Stark. Model-based supervisory control in telerobotics. *Presence: Teleoperators and virtual environments*, 5(2):205–223, 1996.

[20] Kevin Brady and Tsyh-Jong Tarn. Internet-based teleoperation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 644–649, Seoul, Korea, May 2001.

[21] Mattias Bratt, Christian Smith, and Henrik I. Christensen. Design of a control strategy for teleoperation of a platform with significant dynamics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1700–1705, Beijing, China, Oct 2006. IEEE/RSJ.

[22] Mattias Bratt, Christian Smith, and Henrik I. Christensen. Minimum jerk based prediction of user actions for a ball catching task. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2710–2716, San Diego, Ca, USA, Oct 2007. IEEE/RSJ.

[23] M. Buhler and D. E. Koditschek. From stable to chaotic juggling: Theory, simulation, and experiments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1976–1981, Cincinatti, OH, 1990.

[24] G.C. Burdea. Invited review: the synergy between virtual reality and robotics. *IEEE Transactions on Robotics and Automation*, 15(3):400–410, Jun 1999.

[25] E.F. Camacho and C. Bordons. *Model predictive control*. Springer-Verlag, London, 1999.

[26] J. Casper and R.R. Murphy. Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 33(3):367–385, June 2003.

[27] Ho Ching and Wayne J. Book. Internet-based bilateral teleoperation based on wave variable with adaptive predictor and direct drift control. *Journal of Dynamic Systems, Measurement, and Control*, 128(1):86–93, 2006.

[28] Nak Young Chong, Tetsuo Kotoku, Kohtaro Ohba, Kiyoshi Komoriya, Nobuto Matsuhira, and Kazuo Tanie. Virtual impedance based remote tele-collaboration with time delay. In *Proceedings of the IEEE International Workshop on Robot and Human Interaction*, pages 267–272, Pisa, Italy, 1999.

[29] J. W. Clark. The mobot: A fully remote mobile handling device. Technical Memo 627, Nuclear Electronics Lab, Hughes Aircraft Co., Culver City, California, USA, November 1959.

[30] Sue V. G. Cobb, Sarah Nichols, Amanda Ramsey, and John R. Wilson. Virtual reality-induced symptoms and effects (vrise). *Presence*, 8(2):169–186, April 1999.

[31] Christine Connolly. Kuka robotics open architecture allows wireless control. *Industrial Robot: An International Journal*, 31(1):12–15, 2008.

[32] B. Corteville, E. Aertbelien, H. Bruyninckx, and J. De Schutter. Human-inspired robot assistant for fast point-to-point movements. In *Proceedings of IEEE International Conference on Robotics and Automation, ICRA*, pages 3639–3644, 2007.

[33] J.J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley Pub. Co., Reading, 1986.

[34] A. Davids. Urban search and rescue robots: from tragedy to technology. *Intelligent Systems*, 17(2):81–83, March-April 2002.

[35] Joost C. Dessing, C. E. Peper, Daniel Bullock, and Peter J. Beek. How position, velocity, and temporal information combine in the prospective control of catching: Data and model. *Journal of Cognitive Neuroscience*, 17(4):668–686, 2005.

[36] Sascha E. Engelbrecht. Minimum principles in motor control. *Journal of Mathematical Psychology*, 45(3):497–542, 2001.

[37] Andrew H. Fagg, Michael Rosenstein, Jr. Robert Platt, and Roderic A. Grupen. Extracting user intent in mixed initiative teleoperator control. In *Proceedings of AIAA 1st Intelligent Systems Technical Conference*, Chicago, USA, Sep 2004. American Institute of Aeronautics and Astronautics.

[38] W.R. Ferrell. Delay force feedback. *IEEE Transactions on Human Factors in Electronics*, 7:449–455, Oct 1966.

[39] Paul Fitts. The information capacity of the human motor system in controlling the amplitude of movements. *Journal of Experimental Psychology*, 47:381–391, 1954.

[40] Tamar Flash and Ealan Henis. Arm trajectory modification during reaching towards visual targets. *Journal of Cognitive Neuroscience*, 3:220–230, 1991.

[41] Tamar Flash and Neville Hogan. The coordination of arm movements: An experimentally confirmed mathematical model. *The Journal of Neuroscience*, 5(7):1688–1703, July 1985.

[42] Terrence Fong and Charles Thorpe. Vehicle teleoperation interfaces. *Autonomous Robots*, 11:9–18, 2001.

[43] M. Franken, S. Stramigioli, R. Reilink, C. Secchi, and A. Macchelli. Bridging the gap between passivity and transparency. In *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.

[44] U. Frese, B. Bauml, S. Haidacher, G. Schreiber, I. Schaefer, M. Hahnle, and G. Hirzinger. Off-the-shelf vision for a robotic ball catcher. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1623–1629, 2001.

[45] J. Funda, R.H. Taylor, B. Eldridge, S. Gomory, and K.G. Gruben. Constrained cartesian motion control for teleoperated surgical robots. *IEEE Transactions on Robotics and Automation*, 12(3):453–465, Jun 1996.

[46] Andrej Gams and Pierre-AndrÃ© Mudry. Gaming controllers for research robots: controlling a humanoid robot using a WIIMOTE. In *Proceedings of the 17th International Electrotechnical and Computer Science Conference (ERK08)*, 2008.

[47] Tali Gat-Falik and Tamar Flash. The superposition strategy for arm trajectory modification in robotic manipulators. *IEEE Transactions on Systems, Man, And Cybernetics–Part B: Cybernetics*, 29(1):83–95, Feb 1999.

[48] R. C. Goertz. Fundamentals of general-purpose remote manipulators. *Nucleonics*, 10 (11):36–45, 1952.

[49] R. C. Goertz, R. A. Blomgren, J. H. Grimson, G. A. Forster and. W. M. Thompson, and W. H. Kline. The ANL model 3 master-slave electric manipulator - its design and use in a cave. *Transactions of the American Nuclear Society*, 4(2):219–220, 1961.

[50] K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, and J. Wiegley. Desktop teleoperation via the world wide web. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 654–659, May 1995.

[51] S. Graves and R. Volz. Action selection in teleautonomous systems. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 14–19, Aug 1995.

[52] Cheng Guo and Ehud Sharlin. Exploring the use of tangible user interfaces for human-robot interaction: A comparative study. In *Proceeding of the SIGCHI Conference on Human Factors in Computing Systems*, pages 121–130, 2008.

[53] B. Hannaford. A design framework for teleoperators with kinesthetic feedback. *IEEE Transactions on Robotics and Automation*, 5(4):426–434, Aug 1989.

[54] Kensuke Harada, Hitoshi Hasanuma, Katsumi Nakashima, Yoshihiro Kawai, and Hirohisa Hirukawa. Task autonomy for a teleoperated humanoid robot. In *Proceedings of The Eleventh International Symposium of Experimental Robotics (ISER)*, volume 54 of *Springer Tracts in Advanced Robotics*, pages 533–540, Athens, Greece, July 2008.

[55] Christopher M. Harris and Daniel M. Wolpert. Signal-dependent noise determines motor planning. *Nature*, 394:780–784, Aug 1998.

[56] Alexa Hauck, Michael Sorg, and Thomas Schenk. What can be learned from human reach-to-grasp movements for the design of robotic hand-eye systems? In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 2521–2526, Detroit, Michigan, USA, 1999.

[57] Miguel Hernando and Ernesto Gambao. *Advances in Telerobotics*, volume 31 of *STAR*, chapter Teleprogramming: Capturing the Intention of the Human Operator, pages 303–320. Springer-Verlag, 2007.

[58] J.W. Hill, P.S. Green, J.F. Jensen, Y. Gorfu, and A.S. Shah. Telepresence surgery demonstration system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2302–.2307, May 1994.

[59] Sandra Hirche and Martin Buss. Human perceived transparency with time delay. *STAR Advances in Telerobotics*, 31:191–209, 2007.

[60] Sandra Hirche, Manuel Ferre, Jordi Barrio, Claudio Melchorri, and Martin Buss. *Advances in Telerobotics*, volume 31 of *STAR*, chapter Bilateral Control Architectures for Telerobotics, pages 163–176. Springer-Verlag, 2007.

[61] G. Hirzinger, K. Landzettel, and C Fagerer. Telerobotics with large time delays. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 571–578, 1994.

[62] G. Hirzinger, N. Sporer, A. Albu-Schafer, M. Haahnle, and A. Pascucci. DLR's torque-controlled light weight robot iii - are we reaching the technological limits now? In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1710–1716, 2002.

[63] Gerd Hirzinger, Bernard Brunner, Hohannes Dietrich, and Johann Heindl. Sensor-based space robotics — ROTEX and its telerobotic features. *IEEE Transactions on Robotics*, 9(5):649–663, 1993.

[64] Gerd Hirzinger, Max Fischer, Bernard Brunner, Ralf Koeppe, Martin Otter, Markus Gerbenstein, and Ingo Schäfer. Advances in robotics: The DLR experience. *International Journal of Robotics Research*, 18(12):1064–1087, November 1999.

[65] Gerd Hirzinger, J. Heindl, and K. Landzettel. Predictive and knowledge-based telerobotic control concepts. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 1768–1777, May 1989.

[66] Neville Hogan. An organizing principle for a class of voluntary movements. *Journal of Neuroscience*, 4:2745–2754, 1984.

[67] Neville Hogan and Tamar Flash. Moving gracefully: quantitative theories of motor coordination. *Trends in Neurosciences*, 10(4):170–174, 1987.

[68] Peter F. Hokayem and Mark W. Spong. Bilateral teleoperation: An historical survey. *Automatica*, 42:2035–2057, 2006.

[69] B.M. Hove and J.J. Slotine. Experiments in robotic catching. In *Proceedings of the American Control Conference*, volume 1, pages 380–385, Boston, MA, Jun 1991.

[70] Cunjun Huang, Glenn Wasson, Majd Alwan, Pradip Sheth, and Alexandre Ledoux. Shared navigational control and user intent detection in an intelligent walker. In *AAAI Fall 2005 Symposium (EMBC)*, 2005.

[71] C. Sean Hundtofte, Gregory D. Hager, and Allison Okamura. Building a task language for segmentation and recognition of user input to cooperative manipulation systems. In *Proceedings of the 10th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 225–230. IEEE, 2002.

[72] T. Imaida, Y. Yokokohji, T. Doi, M. Oda, and T. Yoshikawa. Ground-space bilateral teleoperation of ets-vii robot arm by direct bilateral coupling under 7-s time delay condition. *IEEE Transactions on Robotics and Automation*, 20(3):499–511, June 2004.

[73] Idaku Ishii and Masatoshi Ishikawa. Self windowing for high-speed vision. *Systems and Computers in Japan*, 32(10):51–58, 2001.

[74] L. D. Jackel, Eric Krotkov, Michael Perschbacher, Jim Pippine, and Chad Sullivan. The DARPA LAGR program: Goals, challenges, methodology, and phase I results. *Journal of Field Robotics*, 23(11–12):945–973, 2006.

[75] Nathanaël Jarrassé, Viviane Pasqui, and Guillaume Morel. How can human motion prediction increase transparency? In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2134–2139, 2008.

[76] E. G. Johnsen and W. R. Corliss. Teleoperators and human augmentation – an AEC-NASA technology survey. Technical Report NASA-SP–5047, Atomic Energy Commission and National Aeronautics and Space Adminstration, Washington, D.C., USA, Jan 1967.

[77] R.E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME — Journal of Basic Engineering*, 82:35–45, 1960.

[78] M. Kawato, Y. Uno, M. Isobe, and R. Suzuki. Hierarchical neural network model for voluntary movement with application to robotics. *IEEE Control Systems Magazine*, 8(2):8–15, Apr 1988.

[79] Mitsuo Kawato. Trajectory formation in arm movements: Minimization principles and procedures. In Howard N. Zelaznik, editor, *Advances in Motor Learning and Control*, Human Kinetics, pages 225–259. Human Kinetics Publishers, Chanpaign Illinois, 1996.

[80] Juha Kela, Panu Korpipää, Jani Mäntyjärvi, Sanna Kallio, Giuseppe Savino, Luca Jozzo, and Di Marca. Accelerometer-based gesture control for a design environment. *Personal Ubiquitous Computing*, 10(5):285–299, 2006.

[81] Hassan K. Khalil. *Nonlinear Systems*. Pearson Education, 3 edition, 2000.

[82] A. Kheddar, C. Tzafestas, and P. Coiffet. The hidden robot concept — high level abstraction teleoperation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1818–1824, 1997.

[83] Abderrahmane Kheddar, Ee-Sian Neo, Riichiro Tadakuma, and Kazuhito Yokoi. *Advances in Telerobotics*, volume 31 of *STAR*, chapter Enhanced Teleoperation Through Virtual Reality Techniques, pages 139–159. Springer-Verlag, 2007.

[84] K. Kosuge, H. Murayama, and K. Takeo. Bilateral feedback control of telemanipulators via computer network. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1380–1385, Osaka, Japan, Nov 1996.

[85] Tetsuo Kotoku. A predictive display with force feedback and its application to remote manipulation system with transmission time delay. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 239–246, Raleigh, NC, Jul 1992.

[86] H.I. Krebs, J.J. Palazzolo, L. Dipietro, M. Ferraro, J. Krol, K. Rannekleiv, B.T. Volpe, and N. Hogan. Rehabilitation robotics: Performance-based progressive robot-assisted therapy. *Journal of Autonomous Robots*, 15(1):7–20, Jul 2003.

[87] Oliver Kreylos. Oliver kreylos' research and development homepage - wiimote hacking, Accessed Sep 15, 2008. URL `http://idav.ucdavis.edu/ okreylos/ResDev/Wiimote/index.html`.

[88] Nadine Krüger. Entwicklung einer humanoiden ball-fang-strategie für ein roboter-hand-arm-system. Master's thesis, Technischen Universität München, 2006.

[89] James Kuffner, Koichi Nishiwaki, Satoshi Kagami, Masayuki Inaba, and Hirochika Inoue. *Robotics Research*, volume 15 of *Springer Tracts in Advanced Robotics (STAR)*, chapter Motion Planning for Humanoid Robots, pages 365–374. Springer Berlin / Heidelberg, 2005.

[90] Jean Kumagai. Techno cops [police robotic and electronic technology]. *IEEE Spectrum*, 39(12):34–39, Dec 2002.

[91] Konstantinos J. Kyriakopoulos and George N. Saridis. Minimum jerk path generation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 364–369, Philadelphia, PA, USA, 1988.

[92] Michael F. Land and Peter McLeod. From eye movements to actions: How batsmen hit the ball. *Nature Neuroscience*, 3(12):1340–1345, Dec 2000.

[93] M. Lapping-Carr, O.C. Jenkins, D.H. Grollman, J.N. Schwertfeger, and T.R. Hinkle. Wiimote interfaces for lifelong robot learning. In *AAAI Symposium on Using AI to Motivate Greater Participation in Computer Science*, Palo Alto, CA, USA, Mar 2008.

[94] Hugh H. S. Liu and Grantham K. H. Pang. Accelerometer for mobile robot positioning. *IEEE Transactions on Industry Applications*, 37(3):812–819, May/Jun 2001.

[95] R.C. Luo and Tse Min Chen. Remote supervisory control of a sensor based mobile robot via internet. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 1163–1168, Sep 1997.

[96] A.J. Madhani, G. Niemeyer, and Jr. Salisbury, J.K. The black falcon: a teleoperated surgical instrument for minimally invasive surgery. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 936–944 vol.2, Oct 1998.

[97] Jacques Marescaux, Joel Leroy, Francesco Rubino, Michelle Smith, Michel Vix, Michele Simone, and Didier Mutter. Transcontinental robot-assisted remote telesurgery: Feasibility and potential applications. *Annals of Surgery*, 235(4):487–492, 2002.

[98] Paul Marks. Wii and iPhone help military control freaks. *New Scientist*, page 26, Mar 29 2008.

[99] Joseph McIntyre, Alain Berthoz, M Zago, and F Lacquaniti. Does the brain model newton's laws? *Nature Neuroscience*, 4(7):693–694, 2001.

[100] Theodore E. Milner. A model for the generation of movements requiring endpoint precision. *Neuroscience*, 49(2):487–496, 1992.

[101] Probal Mitra and Günther Niemeyer. Model-mediated telemanipulation. *The International Journal of Robotics Research*, 27(2):253–262, Feb 2008.

[102] Adrián Mora and Antonio Barrientos. *Advances in Telerobotics*, volume 31 of *STAR*, chapter Re-configurable Control Scheme for Guiding Telerobotics, pages 289–301. Springer-Verlag, 2007.

[103] P. Morasso and F. A. Mussa Ivaldi. Trajectory formation and handwriting: A computational model. *Biological Cybernetics*, 45(2):131–142, Sep 1982.

[104] Brian K. Muirhead. Mars rovers, past and future. In *Proceedings of the IEEE Aerospace Conference*, volume 1, pages 128–134, 2004.

[105] Saghir Munir and Wayne J. Book. Internet-based teleoperation using wave variables with prediction. *IEEE/ASME Transactions on Mechatronics*, 7(2):124–133, Jun 2002.

[106] X. Navarro, T. Kruger, N. Lago, S. Micera, T. Stieglitz, and P. Dario. A critical review of inter faces with the peripheral ner vous system for the control of neuroprostheses and hybrid bionic systems. *Journal of Peripheral Nervous System*, 10:229–258, 2005.

[107] Winston L. Nelson. Physical principles for economies of skilled movements. *IEEE/ASME Transactions on Mechatronics*, 46:135–147, 1983.

[108] G. Niemeyer and J.-J.E. Slotine. Using wave variables for system analysis and robot control. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1619–1625, Albuquerque, NM, Apr 1997. IEEE.

[109] G. Niemeyer and J.-J.E. Slotine. Toward bilateral internet teleoperation. In K. Goldberg and R. Siegwart, editors, *Beyond webcams*, chapter 11, pages 193–213. MIT Press, Cambridge, MA, 2001.

[110] Shuichi Nishio, Hiroshi Ishiguro, Miranda Anderson, , and Norihiro Hagita. Representing personal presence with a teleoperated android: A case study with family. In *Proceedings of the 2008 AAAI Spring Symposium, Session 04, on Emotion, Personality, and Social Behavior*, pages 96–103, 2008.

[111] Julio E. Normey-Rico and Eduardo. F. Camacho. Dead-time compensators: A survey. *Control Engineering Practice*, 16(4):407 – 428, 2008. Special Section on Manoeuvering and Control of Marine Craft.

[112] W.J. Phalen. Investigation of manipulating devices for deep ocean equipment. Technical Note 475, U. S. Naval Civil Engineering Laboratory, Port Hueneme, California, USA, Jan 1963.

[113] Aurelio Piazzi and Antonio Visioli. Global minimum-jerk trajectory planning of robot manipulators. *IEEE Transactions on Industrial Electronics*, 47(1):140–149, Feb 2000.

[114] B. Preising, T.C. Hsia, and B. Mittelstadt. A literature review: robots in medicine. *IEEE Engineering in Medicine and Biology Magazine*, 10(2):13–22, 71, Jun 1991.

[115] M. H. Raibert. *Legged robots that balance*. MIT Press, Cambridge, MA, 1986.

[116] A. A. Rizzi and D. E. Koditschek. Progress in spatial robot juggling. In *Proceedings of IEEE International Conference on Robotics and Automation, ICRA*, pages 775–780, Nice, France, 1992.

[117] Louis B. Rosenberg. The use of virtual fixtures to enhance telemanipulation with time delay. In *Proceedings of the ASME Winter Anual Meeting on Advances in Robotics, Mechatronics, and Haptic Interfaces*, volume 49, pages 29–36, 1993.

[118] A. Rovetta, R. Sala, Xia Wen, and A. Togno. Remote control in telerobotic surgery. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 26(4):438–444, Jul 1996.

[119] Y. Rybarczyk, E. Colle, and P. Hoppenot. Contribution of neuroscience to the teleoperation of rehabilitation robot. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, volume 4, Oct. 2002.

[120] Daisuke Sakamoto, Takayuki Kanda, Tetsuo Ono, Hiroshi Ishiguro, and Norihiro Hagita. Android as a telecommunication medium with a human-like presenceta. In *Proceedings of 2nd ACM/IEEE International Conference on Human-Robot Interaction*, pages 193–200, Washington DC, USA, March 2007.

[121] G. Sandini, G. Metta, and D. Vernon. The icub cognitive humanoid robot: An open system research platform for enactive cognition. In *Artificial Intelligence 50 years*. Springer Verlag, Berlin, 2008.

[122] Emanuele L. Secco and Antonio Visioli Giovanni Magenes. Minimum jerk motion planning for a prosthetic finger. *Journal of Robotic Systems*, 21(7):361–368, 2004.

[123] Jie Sheng and Mark W. Spong. Model predictive control for bilateral teleoperation systems with time delay. In *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, volume 4, pages 1877–1880, May 2004.

[124] Thomas B. Sheridan. Telerobotics. *Automatica*, 25(4):487–507, 1989.

[125] Thomas B. Sheridan. *Telerobotics, automation, and human supervisory control*. MIT Press, 1992.

[126] Thomas B. Sheridan. Space teleoperation through time delay: Review and prognosis. *IEEE Transactions on Robotics and Automation*, 9(5):592–606, Oct 1993.

[127] Thomas B. Sheridan and W.R. Ferrell. Remote manipulative control with transmission delay. *IEEE Transactions on Human Factors in Electronics*, 4:25–29, 1963.

[128] Takaaki Shiratori and Jessica K. Hodgins. Accelerometer-based user interfaces for the control of a physically simulated character. *ACM Transactions on Graphics*, 27(5): 1–9, 2008.

[129] Nabil Simaan, Russell Taylor, and Paul Flint. *Medical Image Computing and Computer-Assisted Intervention — MICCAI 2004*, chapter High Dexterity Snake-Like Robotic Slaves for Minimally Invasive Telesurgery of the Upper Airway, pages 17–24. Lecture Notes in Computer Science. Springer, Sep 2004.

[130] Shahin Sirouspour and Ali Shahdi. Model predictive control for transparent teleoperation under communication time delay. *IEEE Transactions on Robotics*, 22(6): 1131–1145, Dec 2006.

[131] Christian Smith, Mattias Bratt, and Henrik I Christensen. Teleoperation for a ballcatching task with significant dynamics. *Neural Networks, Special Issue on Robotics and Neuroscience*, 24:604–620, May 2008.

[132] Christian Smith and Henrik I. Christensen. Using COTS to construct a high performance robot arm. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4056–4063, Rome, IT, April 2007. IEEE.

[133] Christian Smith and Henrik I Christensen. Constructing a high performance robot from commercially available parts. *Robotics and Automation Magazine*, 16:4, Dec 2009.

[134] Christian Smith and Henrik I. Christensen. A minimum jerk predictor for teleoperation with variable time delay. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5621–5627, Saint Louis, USA, Oct 2009. IEEE/RSJ.

[135] Christian Smith and Henrik I. Christensen. Wiimote robot control using human motion models. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5509–5515, Saint Louis, USA, Oct 2009. IEEE/RSJ.

[136] Christian Smith and Patric Jensfelt. A predictor for operator input for time-delayed teleoperation. *(In review for) Mechatronics, Special Issue on Design Control Methodology*, 2010.

[137] O. J. M. Smith. A controller to overcome dead time. *ISA Journal*, 6:28–33, Feb 1959.

[138] T. H. Song, J. H. Park, S. M. Chung, S. H. Hong, K. H. Kwon, S. Lee, and J. W. Jeon. A study on usability of human-robot interaction using a mobile computer and a human interface device. In *Proceedings of the ACM International Conference on Human Computer Interaction with Mobile Devices and Services (Mobile HCI)*, pages 462–466. ACM, Sep 2007.

[139] Curtis H. Spenny, Dean L. Schneider, and Thomas E. Deeter. *Wiley Encyclopedia of Electrical and Electronics Engineering*, chapter Telerobotics. John Wiley & and Sons, 1999.

[140] Metthew R. Stein and Richard P. Paul. Operator interaction, for time-delayed teleoperation, with a behavior-based controller. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 231–236. IEEE, 1994.

[141] Vytas SunSpiral, Mark B. Allan, Rodney Martin, and Kevin R. Wheeler. Modelling and classifying six-dimensional trajectories for tele-operation under time delay. In *AAAI Spring 2006 Symposium*, 2006.

[142] Michael J. Tarr and William H. Warren. Virtual reality in behavioral neuroscience and beyond. *Nature Neuroscience*, 5:1089–1092, 2002.

[143] Y. Tsumaki, T. Goshozono, K. Abe, M. Uchiyama, R. Koeppe, and G. Hirzinger. Experimental verification of an advanced space teleoperation system using internet. *Journal of Robotics and Mechatronics*, 12(4), 2000.

[144] Y. Uno, M. Kawato, and R. Suzuki. Formation and control of optimal trajectory in human multijoint arm movement. *Biological Cybernetics*, 61:89–101, 1989.

[145] Paul Varcholik, Daniel Barber, and Denise Nicholson. Interactions and training with unmanned systems and the nintendo wiimote. In *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference*, Orlando, Fl., Dec 2008.

[146] Paolo Viviano and Tamar Flash. Minimum-jerk, two-thirds power law, and isochrony: converging approaches to movement planning. *Journal of Experimental Psychology: Human Perception and Performance*, 21(1):32–53, Feb 1995.

[147] M.W Walker and D.E. Orin. Efficient dynamic computer simulation of robotic mechanisms. In *Transactions of the ASME – Journal of Dynamic Systems, Measurement and Control*, volume 104, pages 205–211, 1982.

[148] Carolina Weber, Verena Nitsch, Ulrich Unterhinninghofen, Berthold Farber, and Martin Buss. Position and force augmentation in a telepresence system and their effects on perceived realism. In *Proceedings of the World Haptics Third Joint EuroHaptics conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC)*, pages 226–231, Washington, DC, USA, 2009. IEEE Computer Society.

[149] Greg Welch and Gary Bishop. An introduction to the kalman filter. Technical Report TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC, Apr 2004.

[150] Daniel E. Whitney. State space models of remote manipulation tasks. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 495–507, 1969.

[151] Daniel E. Whitney. Historical perspective and state of the art in robot force control. *International Journal of Robotics Research*, 6(1):3–14, 1987.

[152] D. Yoerger and J.-J. Slotine. Supervisory control architecture for underwater teleoperation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 2068–2073, Mar 1987.

[153] K. Yokoi, M. Kobayashi, H. Hasunuma, H. Moriyama, T. Itoko, Y. Yanagihara, T. Ueno, K., and Ohya. Application of humanoid robots for teleoperations of construction machines. In *IEEE/RSJ IROS Workshop on Explorations towards Humanoid Robot Applications*, 2001.