SUPERMEDIA ENHANCED INTERNET-BASED REAL-TIME TELEROBOTIC OPERATIONS

BY

IMAD HANNA ELHAJJ

A Dissertation

Submitted to

Michigan State University

in partial fulfillment of the requirements

for the degree of

DOCTORATE OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

2002

ABSTRACT

Supermedia Enhanced Internet-Based Real-Time Telerobotic Operations

By

Imad Hanna Elhajj

THE Internet has added a new dimension to the human life. Once the extent of financial possibilities was revealed, Internet-based applications/services expanded to cover every aspect of our lives. A person can virtually trade, learn, play, control and do many other actions over the Internet. However, all Internet-based activities are still referred to as virtual since the experience is still far from being genuine. The main difference between the virtual and real world is the quality of the human experience. This experience is impeded when the Internet is used because of limited human sensory interaction with the remote object and environment.

This research carried out is an attempt to bridge the gap between the real and virtual worlds. This bridge will create the link that human senses need to be remotely coupled with the environment. This research might be the closest technology will get to achieving teleportation in the near future. The basic idea is to feedback sensory information available from the remote machine to the human operator in real-time. This sensory information can relate to any of the human senses: tactile, heat, visual and auditory.

However, coupling the human with the remote environment via the Internet has many challenges. These challenges relate to human-machine interfacing, remote machine stability, operating efficiency and safety. What creates these challenges is the Internet delay characteristics. This delay experienced over the Internet is random and nondeterministic with no simple stochastic model. Other difficulties include the

existence of a human in the control loop and the non-determinism in the environment or the path.

This research presents a general method for the modeling, control and analysis of real-time bilateral teleoperation systems via the Internet. Petri Nets are suggested as a modeling, design and analysis tool, and event-based control theory is modified for the use in the planning and control of bilateral teleoperation systems. Research shows that this method results in a stable system in spite of the existence of nondeterministic time delay. Moreover, the concepts of event-transparency and event-synchronization for event-based telerobotic control systems are developed and analyzed. The implications of these concepts on the system performance are studied. Furthermore, event-transparency and time-transparency are compared. Also presented is the design of event-based control teleoperation systems that satisfy stability, event-transparency and event-synchronization under random time delay conditions. This design method can be applied to a wide range of systems with different environments, human operators, control commands and supermedia (sensory information: video, audio, haptic, temperature and others) feedback. In addition, the concept of tele-coordination is developed and its implications on the performance of multi-robot coordination systems is illustrated.

Several different systems were developed and all the experiments confirmed the theory presented. Supermedia enhanced real-time Internet-based robotic operations were carried out successfully in a stable, event-transparent and event-synchronized fashion.

To my *great* and *loving* Family...Benedicat vos Deus.

ACKNOWLEDGEMENTS

I would like to acknowledge all the invaluable help from Dr. Ning Xi. Without whom this would not have been possible. His mentoring along the years have been enlightening. This research is a result of numerous discussions with Dr. Xi, his keen insight knowledge and his support.

I would like to thank all my committee members: Dr. Hassan Khalil, Dr. Fathi Salam, Dr. Matt Mutka and Dr. Hairong Li. In addition to their invaluable time spent on the committee, their insightful comments and suggestions enhanced the technical soundness of this research.

I also would like to thank Dr. Yun Hui Liu, Dr. Wen J. Li and Dr. Wai Keung Fung from the Chinese University of Hong Kong and Dr. Toshio Fukuda from Nagoya University for their assistance in the many experiments we did over the years. My apologies for keeping you up so many times till the early morning hours.

I am grateful to my friends and colleagues from the Robotics and Automation Laboratory at Michigan State University. Especially, I would like to thank Dr. Jindong Tan for his assistance in the implementation and testing of several of the systems. And for the many technically stimulating conversations. Also I would like to thank Mr. Amit Goradia, Miss Meng-Meng Yu and Mr. BooHeon Song for their great help in implementing many of the systems.

I am thankful for The National Science Foundation (NSF) and The Defense Advanced Research Projects Agency (DARPA) for providing the financial support for this research.

Last but not least, my thanks go to my family: my grandparents, my parents, my brother and my sister. Without their encouragement, support, understanding, sacrifices and love none of this would have been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

7.51

7.52

7.53

7.54

SINC
was
New York
from the
experie
became a
Artho
cists claim
the use a
every hous
and contro
to the Int
other robo
For the
robots, ref
done via a
been on te
and low cos
teleoperatio
However,
hurdle-delay
tinctive char
with. For da
mission. Co
supermedia

CHAPTER 1

INTRODUCTION

SINCE the days of Karel Capek's R.U.R.(Rossum's Universal Robots) play, which was written in 1921, translated to English in 1923 and presented in London and New York, humans have been fascinated with robots. Robot being the word derived from the Czech robota, meaning "servitude, forced labor". However, robotics did not experience a rapid growth until the seventies and eighties when enabling technologies became available.

Although the robotic technology is young but it is growing at a fast pace. Roboticists claim that robots are analogous to computers. In which case, they estimate that the use and availability of robots is going to be wide spread. There will be robots in every house and business. Naturally as the case with computers, the remote operation and control of robots is going to be a highly desirable feature. So eventually, similar to the Internet there will be a *robonet*, where robots are connected to computers, other robots and even humans.

For the time being, technology goes as far as allowing for the remote control of robots, referred to as teleoperation. Teleoperation takes several forms and can be done via any communication medium, wired or wireless. Recently the main focus has been on teleoperation via the Internet. Motivated by the availability, wide spread and low cost of the Internet, many researchers have focused on the Internet-based teleoperation.

However, all real-time Internet-based teleoperation systems face the same main hurdle–delay. Although delay is present in any communication network, it has distinctive characteristics over the Internet that make it more difficult a problem to deal with. For data transmission, delay is not as important an issue as for control transmission. Considering the Internet as an action super-highway, via which control and supermedia (video, audio, haptic, temperature and others) is being transmitted, time

delay becomes an important factor in the stability, efficiency and safety of real-time teleoperation systems.

In the past decade, considerable research has been done relating to control with time delay. However, all this research has several limitations that make it inapplicable to Internet type delay. Also most of this research concentrates on stability [1]-[8] with no considerable work on transparency [9]-[14]. Research even showed that robust stability and transparency can be conflicting design goals in teleoperation systems under time-based control [9]. In addition, synchronization in such systems has not been addressed. Most of the synchronization work found in the literature relate to multimedia synchronization in open loop systems [15]-[20].

This dissertation presents a general framework for the modeling, analysis and control of supermedia enhanced real-time Internet-based teleoperation systems. This approach adopts event-based control to ensure the stability of bilateral teleoperation systems [6]. In addition, the transparency and synchronization of event-based systems are discussed. Event-transparency and event-synchronization concepts are introduced and their advantages are analyzed. Moreover, the modeling of real-time teleoperation systems is presented with Petri Net recommended as the tool for modeling, analysis and design. Design procedures to ensure, stability, event-transparency and event-synchronization are detailed. The advantage of these features is illustrated using the developed concept of multi-robot tele-coordination.

Experimental results of the Internet based teleoperation of several developed systems verify the analysis. In these experiments, supermedia is fed back from the environment in the form of force, video and/or temperature. These are rendered to the operator in their original forms. It is shown that this approach results in a stable, event-transparent and event-synchronized systems regardless of time delay.

The coming subsections will detail the applications and motivations for teleoperation in general and supermedia enhanced teleoperation. The challenges that face

teleoperation and supermedia enhanced teleoperation are also presented. Then the detailed literature relating to all the aspects of this research are surveyed, pointing out their limitations.

1.1 Applications and Motivations

Teleoperation, which is simply the remote control of a machine, is not a new concept. Because of its many benefits, such as increasing the reachability and safety of humans, it has been extensively researched and studied. The interest in teleoperation steams from this technology's many applications some of which are:

1. Tele-medicine(remote checkup and surgery): Despite all the skeptics, this application is getting major attention from the research community and is experiencing significant advances [21]-[23]. The ultimate goal is to have a highly reliable and accurate system that can be trusted with human life. This will allow experts to treat and to operate on patients who are thousands of miles away. Uses range from trauma care and home care (elderly and disabled remote assistance) to physical therapy and specialized surgery.

2. Tele-manufacturing: The ability to use sophisticated and expensive manufacturing facilities by several users around the world is very cost effective [24].

3. Exploration (sea and space): This application became very popular during the NASA Mars Pathfinder Mission [25]. There are various obvious advantages for this application and the importance of teleoperation for space and deep sea exploration is clear, especially in the safety and cost reduction aspects [26] [27].

4. Hazardous material manipulation: One of the first noticed applications of teleoperation is handling hazardous material; such as, radioactive, chemical or explosive substances [28]. For example, teleoperation can be a natural answer for

chemical spill cleanup.

5. Teleoperated games (entertainment): Several teleoperated games are found on the Internet; for example, in Germany there is an interactive railroad that can be operated remotely. The entertainment industry was one of the first to notice the importance of networking thus strived to link players all over the world with networked games. Now there is the possibility to link several players with several machines or robots.

6. Tele-service (monitoring and maintenance): Machines and complete factories can be monitored from hundreds of miles away. These can also be maintained and controlled remotely.

7. Law enforcement: Several law enforcement agencies already own robots that can be teleoperated to replace humans in hazardous situations and environments.

8. Search and recovery: multiple robots can be used to cover unaccessible areas in search and recovery efforts. Several robots were used for assistance in the recovery efforts at "Ground Zero" shortly after the World Trade Center catastrophe [29].

9. Tele-commerce (sales and demonstration): Customers have the ability to remotely test a machine before buying it with no need for traveling or shipping.

10. Education: Several robots are used online for educational purposes whether in a formal or informal setup [30].

11. Imagine! Teleoperation's application is limited by the imagination. The Internet is a living proof that if you "build it, and they will come"; create the technology and people will imagine applications for it [31].

Regardless of the application, the human experience during teleoperation does not quiet compare to direct interaction because of the limited remote sensory capabilities. To enhance the experience, supermedia feedback is suggested in the next section.

1.2 Supermedia Enhancement

From the point of view of an operator's experience, teleoperation is far from being close to the direct operation. This is due to the fact that the operator's remote sensing capabilities are limited, which disconnects the operator from the environment and the task. It was shown that operators' performance is improved in teleoperation when having haptic feedback [4] [32] [33].

"Of or relating to or proceeding from the sense of touch", such or a slight alteration is the definition found for haptic. Derived from the Greek "haptikos" meaning "to touch", haptics is divided into two major categories of sensory information: tactile and kinesthetic. Tactile sensing includes temperature, skin curvature and stretch, vibration, slip, pressure and contact force. It gives a perception of surface textures and geometry. Kinesthetic information relates to the physical forces applied to and by the body. This includes sensing and awareness of body position and orientation, which is also known as proprioception. When haptic feedback is present, the operator is said to be kinethetically coupled to the environment, and the teleoperator is said to be controlled bilaterally.

The most common form of haptic feedback used in teleoperation is force. However, there are many applications which require additional types of feedback for feasibility and safety. An example is classification and identification, where temperature is required to identify certain objects and materials [34] [35]. Also in some medical applications the temperature change of the organ operated on is critical for safety. Therefore, supermedia enhanced teleoperation systems, similar to the one shown in Figure 1.1, are required. Supermedia is used to describe the collection of all the

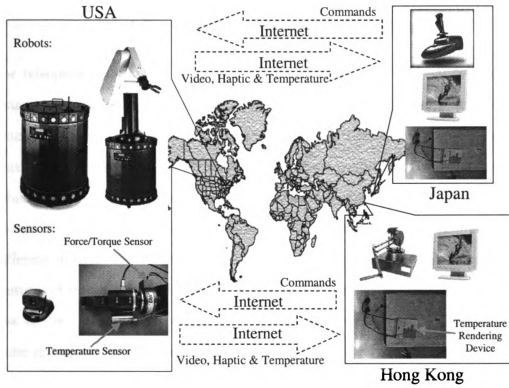

Figure 1.1: The general architecture of supermedia enhanced teleoperation systems.

feedback streams, such as haptic, video, audio, temperature and others. Supermedia enhances efficiency, increases the operation's safety and provides telepresence, which is the human experience of existing in a remote location. Also experiencing supermedia feedback is an unusual and fun experience which makes it a very attractive feature in entertainment applications.

Teleoperation in general, can be carried out using any kind of communication media. But since the Internet had matured into what it is today, interest has been in teleoperating via the Internet. The use of the Internet is motivated by several reasons; in comparison to a dedicated link, the Internet is less expensive (free in many cases), more publicly available, has world wide coverage and does not require any special hardware. However, with all these advantages comes several disadvantages, which make teleoperation via the Internet very challenging. These challenges are the topic of the next section.

1.3 Challenges and Difficulties

For teleoperation systems to become widely acceptable, several safety and performance specifications have to be met. Stability, transparency and synchronization are among the main features that are desired in any teleoperation system. The teleoperation system should be stable under all operation conditions. Transparency, which relates to how realistic the operation is, should be offered. Also synchronization should be present between the operator and the remote machine and between the different supermedia streams. In addition, reliability, safety and efficiency should be features of the teleoperation system. All these features have to be guaranteed regardless of the characteristics of the communication network used. However, ensuring these in teleoperation systems, specifically Internet based ones, is challenging.

The main difficulties relate to the lack of a general design method for such systems. Also the uniqueness in teleoperation control systems is the existence of the human operator in the control loop, which adds to the complexity of the design and analysis process. In addition, teleoperation systems are clearly characterized by nondeterminism in the environment or the path, which makes some assumptions limiting. However, by far the main difficulties are caused by the communication characteristics of such networks especially the Internet.

The Internet does not offer any guarantees for bandwidth or delay. The Internet, in its current form, was not designed and implemented having in mind applications such as teleoperation. Although many changes are being proposed to make the Internet a more diverse communication medium, it still wont be able to guarantee communication parameters.

As any communication link, the Internet introduces delay in the system. This delay can be due to several reasons: propagation, congestion, or low resources. For a communication link between the ground station and a space telerobot in low earth orbit delay is expected to be as long as 2-8 seconds [1] [27] [36]-[38]. The same

amount of

delay is

it could hav

model delay

In [39] a wa

in a stati

It ad

that is why

of all pa

several de

All th

designing

delay can

in several

temperatur

used the sy

addition to

the transpa

solve the ab

of sections 1

1.4

What follow

the problem

presented in

are divided

Wavelet a

amount of delay can be expected for deep sea operations. As for the Internet, the delay is unpredictable, no upper bound can be specified because of packet loss and it could have a very complex stochastic model. Several attempts have been made to model delay over the Internet, and several complex models have been derived [39]. In [39] a wavelet-based model[1] was derived. This delay was shown to be self-similar in a statistical sense [40].

In addition, packets sent over the Internet can be lost and can arrive out of order, that is why TCP/IP communication protocol was used, since it ensures the arrival of all packets in order. More severely, Internet connections can be lost, that is why several design precautions have to be taken.

All these communication features become significant difficulties in the face of designing teleoperation systems that enjoy certain performance features. For instance, delay can render the system unstable if time-based control is used. It was mentioned in several references that delays in the order of a tenth of a second destabilize the teleoperator [1]-[5] [9] [27] [32] [36]-[38] [41]-[45]. When haptic feedback is being used the system was stabilized only when the bandwidth was severely reduced. In addition to instability, desynchronization occurs between the master and slave and the transparency of the system is lost. Several approaches have been suggested to solve the above mentioned problems, and these with their limitations will be the topic of sections 1.4 and 1.5.

1.4 Overview of Existing Methods and Approaches

What follows is a summary of the different methods and techniques used to overcome the problems faced in teleoperation in general. The different approaches will be presented in this section and section 1.5 will cover their limitations. The approaches are divided into two groups. One approach concentrates on the *robotics aspect* by

[1] Wavelet analysis is most used for statistically self-similar processes.

introduction

approach

protocols

loop

the

Most of the

feedback

Control

The problem

Several

developed

In [46]

use shared

operator

for [36],

Model (VIN

control and

introduced a

transmission

separation

introducing control techniques and force/image feedback with prediction. The other approach concentrates on the *communication aspect* by using new communication protocols or by modifying the existing ones, by predicting delay or by studying open loop multimedia communication. Note that no approach combined both aspects in the solution.

1.4.1 Robotics Approach

Most of the approaches in this category relate to control schemes and force/image feedback with prediction. Some of them are a combination of several approaches.

Control

The problem is handled by changes in the controller's modeling, design and algorithm. Several design techniques are used, some of which apply only to the specific system developed and others that are more general.

In [46] [47] an observer for linear feedback control with delays is designed. [1] [27] use shared compliant control, where the control task is shared by both the human operator and the autonomous compliant control of the remote robot system. As for [36], time and position desynchronization was used and in [2] Virtual Internal Model (VIM) was proposed. [3] used a design method based on the $\mathcal{H}_\infty$-optimal control and μ-synthesis framework. The work of Anderson and Spong in [4] and [32] introduced a controller that was derived from modeling the network as a loss-less transmission line. Another method was based on stochastic control theory and a separation property [43].

Sensory

This app
back elem
the techn

The m
task spac
The other
robot is f
usually us
the positio
is then disp
Predicted f
just display

The commun
after the und
performance
the network,
than to close

In the fir
cation proto
test, since th
components
developed a
an application
running at th

Sensory Feedback

This approach includes force/torque feedback and video feedback. Any of these fed back elements can be either real or virtual. Therefore, in the virtual case, some of the techniques use prediction.

The most common feedback in teleoperation is video, where cameras capture the task space and then these images are sent back to the operator [1] [27] [37] [38] [41]. The other sensory feedback is usually force or torque, where the force sensed by the robot is fed back to the operator [4] [5] [32] [44] [45]. Although these two techniques usually use real values, sometimes prediction is employed in parallel. For example, the position of the manipulator can be predicted using a simulation program and it is then displayed with the real image so that the operator can compare [38] [41] [42]. Predicted force can also be used, where the simulated value is sent to the master or just displayed on the monitor [42].

1.4.2 Communication Approach

The communication approaches can be categorized in three main groups; ones which alter the underlying network or communication protocol in order to satisfy certain performance conditions, ones which attempt to predict the delay or the performance of the network, and ones which are more related to open loop multimedia communication than to closed loop teleoperation.

In the first category, approaches try mainly to replace the underlying communication protocol or at least modify it. These methods are difficult to implement and test, since the routing algorithm and communication protocol used by all the different components in the communication system would have to be modified. Note that none developed a method that does not require major system modification; for example, an application level solution, where the routine that is handling the delay problem is running at the application level.

One of the approaches proposes using rate-based flow control, where the rate at which the user is allowed to transmit is determined by the switches on the virtual circuit between the source and destination [48] [49]. Another approach used dynamic congestion control and error recovery algorithm over a heterogeneous Internet [50].

In the second category, many attempts are being made to predict the delay over the Internet specifically for teleoperation purposes. However, because of the complexity of the models, very high levels of accuracy are not attainable. Even highly accurate predictions are not acceptable if they do not guarantee %100 accuracy, because in teleoperation less than %100 stability is not acceptable. In addition, some of those approaches consider the delay to be the same in forward and feedback communication branches, or simply they predict the round trip delay or they do not scale or adapt to changes well [51] [52].

In the last category, the main work done is related to the synchronization of multimedia over the Internet [15]-[20]. This research considers open loop systems, where control does not exist. The main topics are the synchronization of different fed back streams such as video and audio. The methods developed in this category can not be adapted as is because they do not consider control and stability issues.

1.5 Limitations of Existing Solutions

Most of the methods used, share some disadvantages that will be discussed in this section. By far the most common limiting assumption, is taking delay as having an upper bound. An upper bound means taking a value for the delay beyond which the system would either become unstable or has to stop operating. For example, the analysis and implementation might assume that the delay is less than 3 seconds, and if it happens that the delay actually exceeds 3 seconds the system either becomes unstable or has to stop operating. Another limitation is not analyzing or testing with nondeterministic time delays. Also some approaches assume that the delay in the

communication loop is the same in both ways. All of these assumptions are limiting when the Internet in involved, since no upper bound can be obtained because of the possibility of packet loss and the delays are proven to be "random" and "non-deterministic" [39] [40].

Moreover, some methods are based on a specific model of the environment and/or the operator, which is a limiting factor considering the complexity of modeling a human and considering that teleoperation is usually in an environment unknown a priori. As for the predictive methods, they require significant computations and they do not function well with uncertainties in the environment. The communication proposed fixes, are difficult to implement and test since they require major changes in the existing systems.

1.6 Previous Internet Teleoperation and Their Limitations

This section introduces the previous work done specifically in Internet teleoperation and discusses their limitations. Several Internet based robots have been developed and studied, presented here are some examples that cover most of the categories. To clarify a misconception regarding Internet robots, many researchers consider Internet robots to be the ones which are web based, meaning the ones that are controlled via a web page. Many discussions have taken place[2], and the general trend is to consider Internet robots as being the ones controlled over the Internet in any fashion and not only via a web page.

Examining the literature, Internet-based robots can be divided into three categories according to the method used to send commands and receive haptic feedback, and according to the nature of feedback supplied. The categories are: teleprogrammed, telesimulated, and real real-time teleoperated.

The first category, teleprogrammed Internet-based robots, requires the operator

[2]IEEE International Conference on Robotics and Automation, Internet Robotics Workshop, 2000.

to upload a plan or set of commands for the robot to execute. This uploaded program is executed by the robot either autonomously or semi-autonomously. This category can be further divided into "program and watch" robots or "program, watch and intervene" robots. Program and watch robots are ones that execute the program without allowing the operator to change the commands on the fly. Program, watch and intervene robots have the added option of the operator changing the program on the fly before the robot finishes executing it. Typically, teleprogrammed robotic systems include only visual feedback in the form of video.

Some of the most popular robots in this category are:

- Tarn and Brady implemented a semi-autonomous telerobot, where the teleoperator supplies a trajectory for the robot to execute. Then the operator intervenes only in case of unexpected circumstances [53].

- Pai made an expensive experimental facility, the ACME, available via the Internet, where a user could conduct customized measurements and built accurate models from anywhere [54].

- Simmons introduced the autonomous mobile robot Xavier that can be commanded to go to different locations and do different tasks. The commands are high level commands and the only feedback is in the form of video [55].

The second category, telesimulated Internet-based robots, includes systems that feed forward commands in real-time but the feedback is simulated. So the operator sends commands in real-time however the feedback supplied to the operator is simulated. This simulated or predicted feedback is in one of two forms; either it is completely simulated based on a robot and environment model, or it is partially simulated based on a model and corrected by actual feedback. Generally feedback is in the form of predicted display and/or predicted force.

• C
su
• P
is
• S
for
wi
In
The th
feed forwa
The feedb
• Pum
pain
• Khep
feedb
• WebP
via th
Other w
obotics Prog
asks, rangi
iews.
All these
in this disser
[3]http://rami

- Chong developed a system where the operator controls a manipulator and is supplied by overlapping real and predicted display [56].

- Penin introduced a manipulator that can be teleoperated, where the operator is presented with predicted display and force [57].

- Several other systems exist that are based on predictive display and predictive force; however, most of these systems relate to space and underwater robots, which are controlled via a designated communication link and are not considered Internet-based robots.

The third category, real real-time teleoperated Internet-based robotic systems, feed forward commands in real-time and feedback real real-time sensory information. The feedback comes in several forms, the most typical of which are video and force.

- PumaPaint allows the operator to control a Puma 760 in order to make a painting. The control is in real-time and the feedback is only visual [58].

- KhepOnTheWeb is a mobile robot that can be controlled in real-time and video feedback can be obtained [59].

- WebPioneer is a system that allows the teleoperation of a Pioneer mobile robot via the web.

Other web robots have been introduced, and currently the NASA Space Telerobotics Program web site[3] lists more than 20 of them. These robots perform different tasks, ranging from manipulating objects and navigation to manipulating camera views.

All these systems have several limitations and differ from the systems presented in this dissertation in several aspects. First and most important is that these systems

[3]http://ranier.oact.hq.nasa.gov/telerobotics_page/realrobots.html

do not have real-time feedback. Meaning that the only feedback the operator has is visual, where either video or sensory information is sent back to be displayed. This is why these systems can be considered to be open loop control systems. Whereas in the systems discussed in this document, the loop is closed once haptic feedback is included, and therefore it is considered a closed loop control system. Other limitations are assumptions concerning time delay, no research was found dealing with real-time control with haptic feedback in the presence of nondeterministic time delay.

It is worth noting that most of the research found in the literature and discussed in this dissertation for teleoperation under time delay relates to stability, with only few relating to transparency and none relating to synchronization.

1.7 Outline

Outlined here are the remaining parts of the dissertation:

1. Chapter 2 examines the Internet characteristics when viewed as a real-time control link. The Internet characteristics are studied and their effects on feedback systems are discussed. The "randomness" or "non-determinism" of time delay and the instability that it causes are presented.

2. Chapter 3 gives an introduction to the event-based control, which uses an event-based and not a time reference. Then these systems are analyzed in terms of stability and in terms of event-transparency and event-synchronization, which are two concepts developed as part of this research.

3. Chapter 4 details the models of the different systems developed. Also it proposes Petri Net as a modeling, analysis and design tool for event-based teleoperation systems.

4. Chapter 5 Includes the design procedures required to ensure stability, event-

transparency and event-synchronization for teleoperation systems. A Petri Net design methodology resulting in event-synchronous systems is given.

5. Chapter 6 develops the concept of tele-coordination, where multi-robots are teleoperated by multi-operators at different remote locations. This chapter presents the design procedure required to achieve certain levels of coordination.

6. Chapter 7 provides the details of the implementation and experimentation of several systems developed. Details of the hardware and software used in the experiments are given. Also, the properties of the Internet connections used in these experiments are presented. In addition, a detailed discussion about the different results obtained is included.

7. Chapter 8 gives a brief summary and concluding remarks. Also it includes recommendations for future work and research directions.

THE

During the
tion and r
and robot
Internet ha
tance. Rob
and many
what many
worldwide
Examin
merge thes
interesting
to the opera
So, despi
motion lin
machines a
real-time te
conditions r
lem is that t
the properti
and their eff

2.1

First discuss
ering the con

CHAPTER 2

THE INTERNET: REAL-TIME CONTROL MEDIA

During the past decade, two technologies, fed by the great advances in computation and networking, matured into giant interdisciplinary scientific fields. Internet and robotics are two technologies that will greatly influence the human future. The Internet has already secured its place in our lives and proven its indispensable importance. Robotics is yet to achieve that; while major advances have been accomplished and many robotic applications have emerged, it is clear that robots are not yet all what many roboticists predict they will be. Nevertheless, this field is experiencing worldwide interest from scientists, researchers and industries.

Examining this potential, it was not surprising that many researchers are trying to merge these two phenomena and develop Internet-based robotic teleoperation. More interestingly, is Internet-based bilateral teleoperation, where supermedia is fed back to the operator in order to increase efficiency and achieve telepresence.

So, despite the many challenges it presents, the use of the Internet as a communication link for telerobotics is not surprising. The Internet can connect any two machines anywhere in the world or in space, that is why it is a natural media for real-time teleoperation. But once real-time control is considered there are several conditions required to ensure the stability and robustness of the system. The problem is that the Internet does not satisfy many of these conditions. In the next section the properties of the Internet from a real-time control perspective will be presented and their effects on the teleoperation system will be examined.

2.1 Characteristics of Internet Communication

First discussed are the services that the Internet provides for teleoperation. Considering the communication protocol TCP/IP the Internet can provide a *reliable* com-

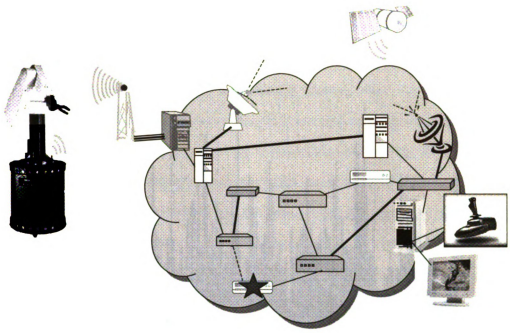

Figure 2.1: An illustration of the Internet topology

munication link. Reliable means that the link does not lose packets nor rearrange them. Many protocols do not offer this, but when TCP/IP is used it implies that the packets are given in order and that no packets are lost (as long as no disconnection occurs). The use of TCP/IP is not limiting since a similar result can be obtained by using any other communication protocol while having the application guarantee that no packets are lost or rearranged.

What the Internet does not provide is a guarantee of service. In other words, the Internet does not provide a permanent communication link, since the connection may be lost at any time. Nor does it provide a guarantee of the quality of service–the Internet does not offer guarantees on communication parameters, such as bandwidth and delay.

This lack of guarantees is a result of the Internet being designed and implemented as a general communication medium, where applications, such as teleoperation, were not considered. Also, the lack of guarantees is an intrinsic property of the Internet, which becomes clear when its topology is considered. Figure 2.1 depicts a typical

Figure 2
ethernet

Internet
ools. Th
and con
can go th
qualities.
advances
that does

So no
the Interne
time and
over the In
times of pa
2.2 gives a

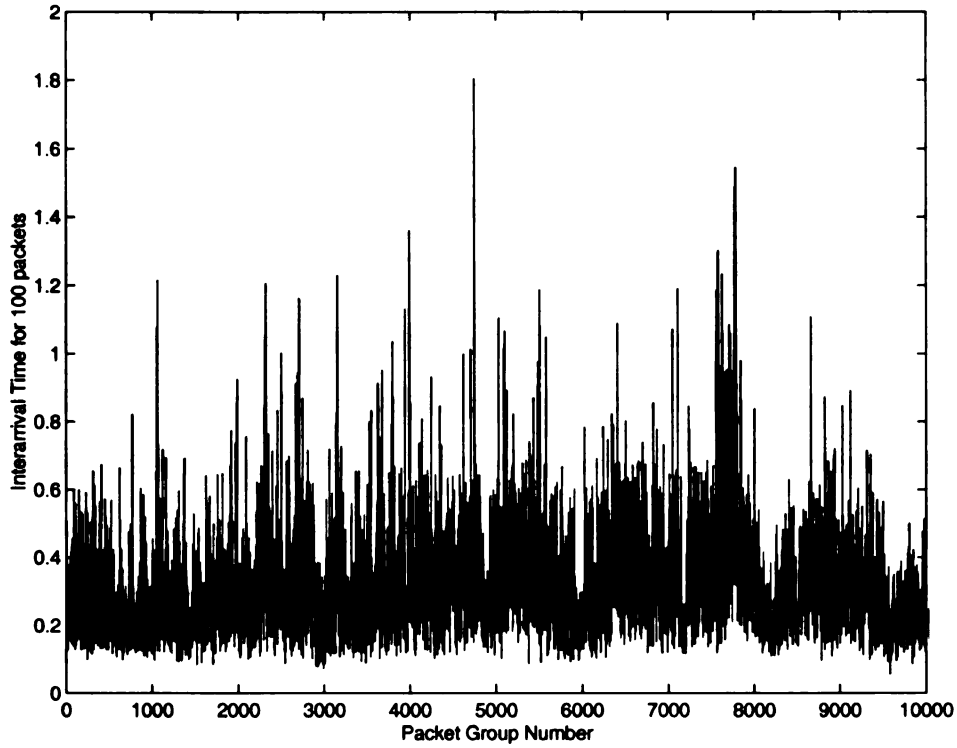

Figure 2.2: Inter-arrival times of groups of packets on the August 1989 Bellcore ethernet trace [39] [40].

Internet model, which consists of heterogeneous hardware and communication protocols. This model includes wired and wireless networks with different types of routers and communication links. So packets traveling between two hosts via the Internet can go through diverse types of routes offering completely different communication qualities. Even with the current suggested changes in the Internet and the many advances in the Internet protocols and topologies, it will always be a diverse network that does not offer assurances regarding the quality of service.

So no matter how good the local area network is, once the communication is over the Internet the link is subject to all kinds of risks. Disconnection can occur at any time and indefinite random delay can be faced; there is no upper bound on the delay over the Internet because of the possibility of packet loss. Therefore, the inter-arrival times of packets cannot be estimated. For an example of this inter-arrival time Figure 2.2 gives a graph of inter-arrival times of groups of packets on the August 1989 Bellcore

ethernet trace [39] [40]. This figure gives an idea of the randomness of the delay over the Internet. Several studies have been made to model this delay and the traffic over networks. These studies have showed that network traffic often possess long-range-dependency (LRD) and can be modeled using fractal and multiplicative multifractal processes [39] [40] [60]-[63]. However, it is not possible to predict the delay based on these statistical models with %100 accuracy. These predictions are done with certain probabilities. The accuracy provided is not sufficient for teleoperation applications, since it is not acceptable, for instance, to guarantee stability with a probability less than 1. Even delay prediction studies, which are devised specifically for Internet-based teleoperation, have their limitations in addition to the lack of accuracy. Some consider the delay to be the same in forward and feedback communication branches, or simply they predict the round trip delay or they do not scale or adapt to changes well [51] [52]. This existence and randomness of delay in Internet communication has significant effects on the real time control, and more so when supermedia feedback is involved.

2.2 Effects of Random Time Delay on Real-Time Control

The main effects of these communication characteristics are: instability, loss of transparency and desynchronization. In 1966, the first work dealing with force feedback under time delay was done (Ferrell 1966). It was shown that delays on the order of a tenth of a second destabilize the teleoperator. It is worth noting that this is true for Internet teleoperation, where the delay faced ranges from 100 milliseconds to a few seconds, and for space and deep sea teleoperation, where the delay is more or less fixed. However, the non-determinism of delay over the Internet introduces additional difficulties. Therefore, teleoperation systems via the Internet will be unstable unless a fix is found.

In addition, perfectly transparent teleoperation in practice is not possible, espe-

cally

depend

have

the time

devices

to a co

certain

depend

chroniza

increase

These

however,

tation.

of event

systems

results in

The

systems.

event-tra

cially under delay conditions such as these encountered over the Internet [9]. As for desynchronization, it is the most intuitive effect of delay, since the operator would have no idea regarding the current robot state. For example, with visual feedback, by the time the operator sees an image it would be an *old* image of the robot. Making decisions based on the old image would be risky and would cause the system to go to a completely different state. So while the operator thinks that the robot is in a certain location it would have already moved to a new one, causing the two to be desynchronized in time and space. Also, if multiple supermedia are fed back, desynchronization between them will occur. This adds to the operator's confusion and increases the possibility of errors.

These delay effects are pronounced in time-based controlled teleoperation systems; however, they are not present in the systems designed and analyzed in this dissertation. The reason for the immunity of the developed systems to delay is the use of event-based control. Event-based control for supermedia enhanced teleoperation systems is developed and analyzed in this dissertation. This control methodology results in stable, event-transparent and event-synchronized systems.

The next chapter introduces event-based planning and control for teleoperation systems. The stability of such systems is also analyzed. In addition, the concepts of event-transparency and event-synchronization are developed and analyzed.

EV

Event
and ana
contr
systems
is intro
under ra
event-sy

3

The ins
time-ba
ent sys
sampl
used th
contro
which
functi
and th
refere
based
event-b

Th
a map

CHAPTER 3

EVENT-BASED PLANNING AND CONTROL FOR REAL-TIME TELEOPERATION

Event-based planning and control for real-time teleoperation systems is introduced and analyzed in this chapter. A comparison of this approach with the time-based control approach is made. The analysis of the stability of event-based controlled systems under random delay conditions is done. In addition, event-transparency is introduced and its implications on the performance of real-time control systems under random non-deterministic delay conditions are analyzed. Also, the concept of event-synchronization is developed.

3.1 Event-Based Real-Time Planning and Control

The instability, lose of transparency and desynchronization caused by time delay in time-based control systems are a result of using time as a reference by the different system components. In other words, the control and feedback signals are being sampled with respect to time. Therefore, intuitively if a non-time based reference is used these effects of delay would be eliminated or reduced significantly. In traditional control systems, the dynamics of the system are modeled by differential equations in which the reference is the time variable, t. In addition, the trajectory is usually a function of time. The general idea of non-time based control is to model the system and the trajectory as functions of a non-time based variable, which is called *motion reference or action reference.* It usually is denoted by s and also called the event-based action reference [64]-[66]. The planning and control of the time-based and the event-based schemes are shown in Figure 3.1.

The Action Reference block in Figure 3.1 acts like a clock for the system. It is a map from the output or state of the robotic system y to a scalar variable s. The

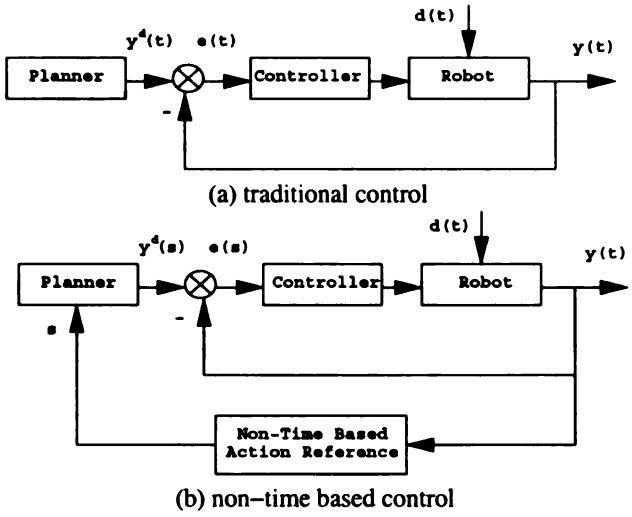

Figure 3.1: The comparison between traditional time-based and event-based planning and control.

robot's plan as well as the operator's commands can be described as functions of s. This reference is usually taken to be a physical output of the system such as distance or position. However, the event-based reference can also be a virtual one that does not correspond to any physical quantity. For example, it can be chosen to be the number of control cycles the system has performed.

Event-based planning and control, which was first introduced in [67], has been used in many studies and several robotic applications [65], and has been proven to be very efficient in dealing with uncertainties and delay. This planning and control method was extended to bilateral teleoperation applications in [6] [68]-[70]. These extensions introduced some modifications over the initial event-based theory in order for it to be applicable for real-time teleoperation systems. The first main consideration is the fact that in teleoperation there is no predefined path. The path is generated in real-time by the operator based on the feedback received. The other issue is that the operator has limited computational power so the operation of the system under event-based control should be transparent to the operator. In addition, the control method has to be independent of the operator and the environment models, and it should not require significant overhead.

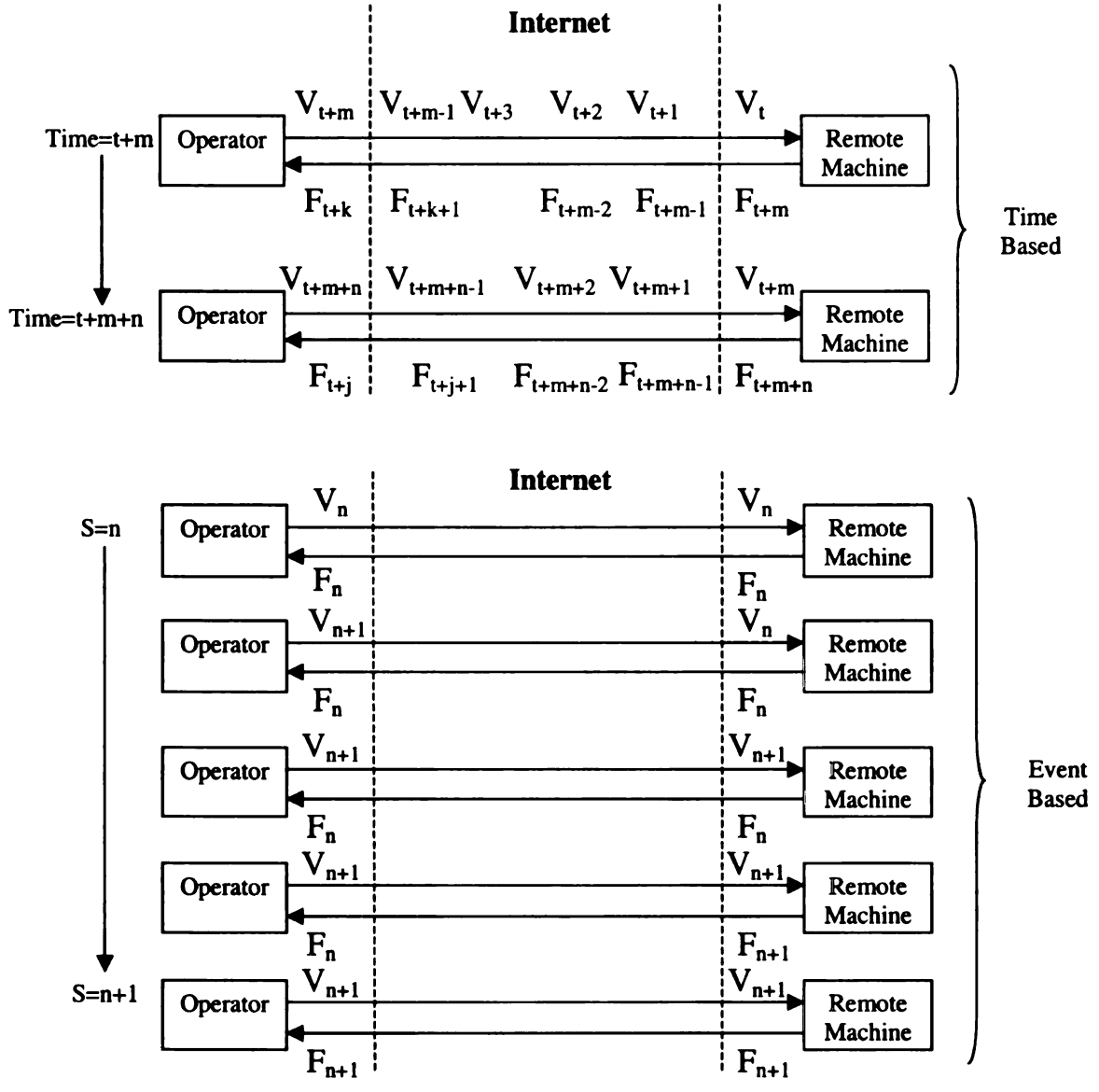

Figure 3.2: Illustration of the buffering effect of delay in time-based planning and control, and the elimination of this effect in event-based planning and control.

Therefore, the event-based reference, s, has to be carefully chosen such that no overhead is required from the operator. In addition, this reference has to be designed in such a way to eliminate the buffering effect of delay. This effect is shown in the time-based part of Figure 3.2, where it is clear in the time-based approach that the delay will cause many signal instances to be flowing within the network. So by the time a new command, V_{t+m}, that was generated as a result of a force, F_{t+k}, arrives at the destination, the state of the machine would have changed due to the commands

that were already buffered in the network, $V_{t+1}, V_{t+2}, \ldots V_{t+m-1}$. This implies that when the robot gets the command V_{t+m} at time $t + m + n$, it has already changed state and is being subject to the force F_{t+n+m}; therefore, the command would be a wrong one since it was a result of F_{t+k} and not F_{t+n+m}. This will cause instability, lose of transparency and desynchronization.

In order to ensure that the right command is being received and executed the buffering effect should be eliminated. The communication and control model should be similar to the one shown in the event-based part of Figure 3.2. As shown, the transmission of new commands and forces is not specified by time. A new command, V_{n+1}, will not be generated and sent until the most up-to-date state of the robot, F_n, is received. At the same time, a new force, F_{n+1}, will not be fed back until a new velocity, V_{n+1}, is received. This implies that there are no commands or forces within the network flowing thus the buffering effect is eliminated. Therefore, the use of event-based planning and control provides the teleoperation system the capability to cope with uncertainties and delays in real-time, without the need to re-plan or re-synchronize [64]-[66].

3.2 Stability Analysis

Several methods have been used to analyze the stability of teleoperation systems. Scattering theory [4] [32] and wave variables [5] are among the main ones. The main concept used in their analysis is based on proving the passivity of the system. A passive system can not create energy, and thus from a control point of view a passive system can not go unstable [4]. The approach presented in this research is slightly different. The main difference is the use of event-based control. Concerning the stability of the system under event-based referencing, the following theorem was proven in [64]-[66]:

Theorem 1 *If the original robot dynamic system (without remote human/autonomous controller) is asymptotically stable with time t as its action reference; and the new non-time action reference, $s=\prod(y)$ is a (monotone increasing) non-decreasing function of time t, then the system is (asymptotically) stable with respect to the new action reference s.*

The only requirement is that the robot is a stable system, which means that the original robot dynamic system (without remote human operator) is asymptotically stable with time, t, as its action reference. This would allow the use of Theorem 1 and proves the (asymptotical) stability of the system with respect to the new action reference s, simply by proving that the new non-time action reference is (monotone increasing) non-decreasing function of time, t. The advantage of this approach, which differentiates it from the other approaches in the literature, is that stability is proven independently of the specific human operator or the statistics of time-delay. It applies to a wide range of systems operating in unknown environments.

Therefore, stability is proven for event-based controlled teleoperation systems regardless of the time delay and the specifics of the system components, as long as it is shown that the action reference is non-decreasing function of time.

As for transparency and synchronization, they are relative performance measures of teleoperation systems. The current technology does not allow for perfect transparency and synchronization in Internet based teleoperation if time-based control is used. The main limitations, which are communication related, are bandwidth and time delay. These limitations are augmented in the Internet since it does not offer any guarantees on bandwidth or delay. Especially, that the Internet in its current form, was not designed and implemented having in mind applications such as teleoperation. Therefore, there is a need for teleoperation systems to offer consistency in the performance under the dynamic communication conditions encountered, which is the topic of the next section.

3.3 Event-Transparency Analysis

In the time domain, transparency is simply a measure of how closely the feel of interaction with the environment in teleoperation resembles the actual feel when the operator directly interacts with the environment. Formally, perfect time-transparency is satisfied under the condition

$$Z_t(t) = Z_e(t) \tag{3.1}$$

where $Z_t(t)$ is the transmitted or "felt" impedance and $Z_e(t)$ is the slave environment impedance. Impedance is defined as the ratio of force to velocity. Therefore,

$$Z_t(t) = \frac{F_h(t)}{V_h(t)} \qquad Z_e(t) = \frac{F_e(t)}{V_e(t)} \tag{3.2}$$

where F_h and V_h are the force and velocity generated by the joystick respectively; similarly, F_e and V_e are the slave force and velocity respectively. It is intuitive that under time-based control perfect time transparency in teleoperation is not possible because of delay [9]. However, in event-based control some type of consistency in the control can be achieved. This consistency is event-transparency and is defined as,

Definition 1 *A control system is event-transparent if,* $\frac{F_h(s)}{V_h(s)} = C\frac{F_e(s)}{V_e(s)}$ *and* $F_e(s)_{rd} = F_e(s)_{nd}$*, where* C *is a constant,* s *is the action reference,* $F_e(s)_{rd}$ *and* $F_e(s)_{nd}$ *are the supermedia sensed by the system under random delay and no delay conditions respectively.*

It is clear that this generalized definition can apply to any control system, where the F's are the feedback and the V's are the control signals. In the special case of teleoperation systems, F is usually force or some type of supermedia and V is the operator's command. To design event-transparent systems, first design the system to

be an event-based control system. That is the signals in the system are being sampled uniformly with respect to a reference, s, other than time. In order to ensure stability of the teleoperation system, this action reference has to be a non-decreasing function of time [6]. Being an event-based control system gives

$$F_h(s) = \alpha F_e(s) \quad and \quad V_h(s) = \beta V_e(s) \tag{3.3}$$

$$\Rightarrow \frac{F_h(s)}{V_h(s)} = C\frac{F_e(s)}{V_e(s)}$$

where α, β and C are simply scaling factors. In addition, s has to ensure that $F_e(s)_{rd} = F_e(s)_{nd}$. The choice of s depends on the type of feedback being used.

From the definition, it implies that if a system is event-transparent then regardless of random time delay, the control received under random time delay as a function of the action reference, $V_e(s)_{rd}$, is equal to the control received by the robot under no time delay as a function of the action reference, $V_e(s)_{nd}$. This is deduced from the fact that the choice of s ensured that the haptics detected by the robot under time delay as a function of the event, $F_e(s)_{rd}$, are equal to the haptics detected by the robot under no time delay as a function of the event, $F_e(s)_{nd}$.

This means that if the event-based reference is chosen in such a way as to make the forces equal in both cases then the control will also be equal. Thus the behavior of the operator will be consistent with respect to the event-based reference and the system will be event-transparent. Therefore, the delay will have no effect on the control signal with respect to the event-based action reference. Obviously, in the time-based case this property is not true.

To illustrate the implications of event-transparency and to compare it with time-transparency, Figure 3.3 shows the sampling of the different signals under different

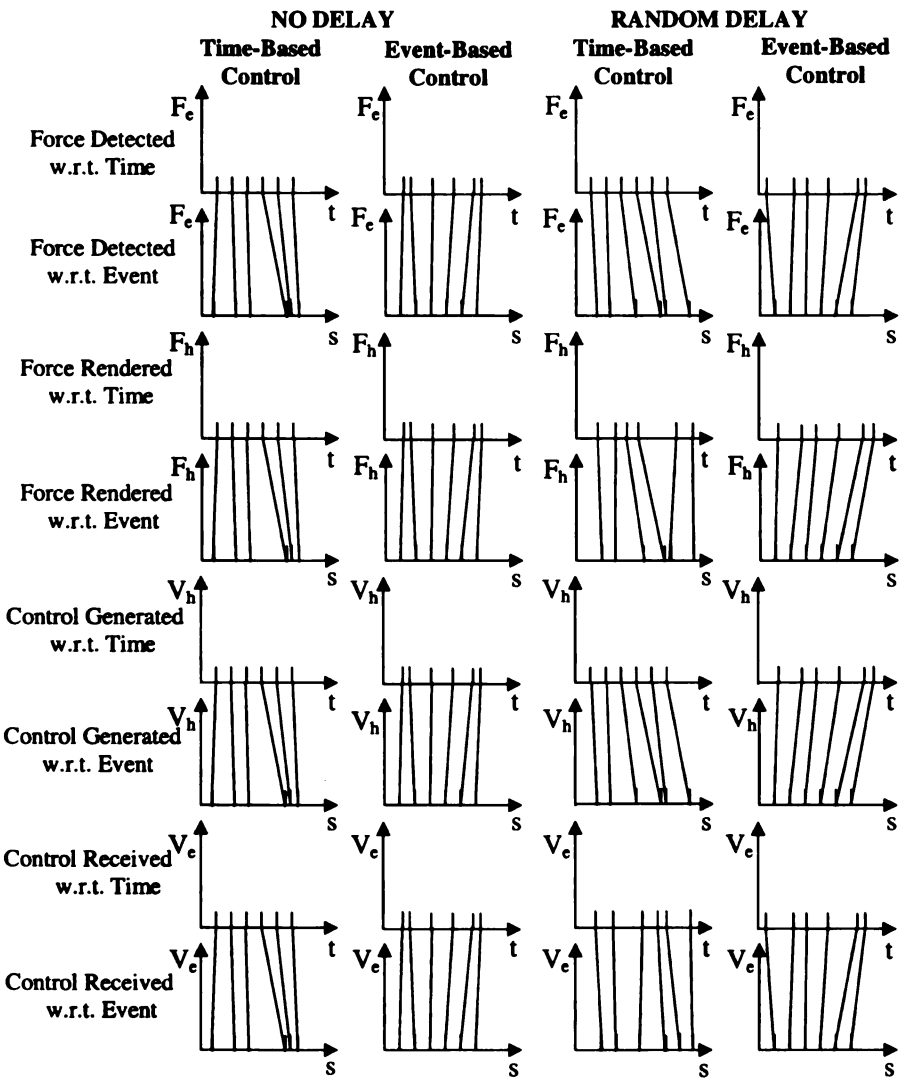

Figure 3.3: Sampling of signals under different conditions.

conditions. In the first column are the signals in the time-based control case and in the second column are the signals in the event-based control case both under no delay conditions. In the third column are the signals in the time-based control case and in the fourth column are the signals in the event-based control case both under random delay conditions. The first and second rows are the force detected or sensed by the robot, F_e, with respect to time and event references respectively. The third and fourth rows are the force rendered by the joystick or applied on the operator, F_h,

with respect to time and event references respectively. The fifth and sixth rows are the operator's control or commands generated, V_h, with respect to time and event references respectively. The seventh and eighth rows are the control received by the robot, V_e, with respect to time and event references respectively.

It is clear in the first column that the sampling is uniform with respect to time for all the signals. Although not uniform with respect to the event-based reference but is the same for all the signals. The opposite is true for the second column, that is the sampling is uniform with respect to the event-based reference and non-uniform but the same with respect to time. This uniformity in the sampling of the transmitted and received signals (F_e and F_h, V_h and V_e) and in the sampling of the received feedback, F_h, and the generated control, V_h, ensures that the control being received by the robot corresponds to the feedback generated at the same reference (time or event-based control). This property translates into stability, transparency and synchronization for the system under no delay conditions.

However, in the random delay case, time-based control loses this uniformity in the sampling as seen in the third column, because the random delay is causing non-uniformity in the arrival of signals ($F_h(t)$ and $V_e(t)$) with respect to time. This implies that the control received by the robot, $V_e(t)$, does not correspond to the feedback generated, $F_e(t)$, at the same time instant. Therefore, desynchronization occurs causing instability and the loss of transparency. On the other hand, event-based control overcomes this problem by sampling the signals with respect to a reference or parameter other than time. As column four illustrates, F_h and V_h with respect to time and event are sampled at the same reference values. Also, examining time-based control with delay shown in column three, it is clear that not all the signals are uniformly sampled with respect to any one reference. Whereas, in event-based control with delay shown in column four, all the signals are uniformly sampled with respect to the event-based reference. This uniformity in sampling results in $F_h(t)$ and $V_h(t)$ being sampled at

the same time instances as seen in column four, which results in the consistency of control under random delay conditions. This consistency can be formally stated with the following theorem,

Theorem 2 *If the event-based action reference, s, of an event-based control system is chosen such that $F_e(s)_{rd} = F_e(s)_{nd}$ then $V_e(s)_{rd} = V_e(s)_{nd}$, where $F_e(s)_{rd}$ and $F_e(s)_{nd}$ are the supermedia sensed by the system under random delay and no delay conditions respectively, and where $V_e(s)_{rd}$ and $V_e(s)_{nd}$ are the control received under random time delay and no time delay conditions respectively.*

Proof. First it will be shown that $F_h(t)$ and $V_h(t)$ are being sampled at the same time instances. To prove this, F_h and V_h with respect to time and event references are analyzed for event-based control and time-based control under time delay conditions. In event-based control, assume that the deviation of the nonuniform samples from a set of uniform samples is represented by δ_{1n} and δ_{2n} and the event sampling periods are S_1 and S_2. So the samples are $F_h(nKS_1 - \delta_{1n})$ and $V_h(nKS_2 - \delta_{2n})$, where K is a scaling factor. In order to determine the relationship between the sampling instances of signals $F_h(t)$ and $V_h(t)$, the signals $\theta_1(s)$ and $\theta_2(s)$ are formed [71] [72]:

$$\theta_1(s) = \sum_{n=-\infty}^{\infty} \delta_{1n} \frac{\sin w(s - nKS_1)}{w(s - nKS_1)} \tag{3.4}$$

$$\theta_2(s) = \sum_{n=-\infty}^{\infty} \delta_{2n} \frac{\sin w(s - nKS_2)}{w(s - nKS_2)} \tag{3.5}$$

Sample values of $\theta_1(s)$ and $\theta_2(s)$ are δ_{1n} and δ_{2n}.

$$\theta_1(nKS_1) = \delta_{1n} \quad \textit{and} \quad \theta_2(nKS_2) = \delta_{2n} \tag{3.6}$$

Let

$$t_1 = s - \theta_1(s) \tag{3.7}$$

$$t_2 = s - \theta_2(s) \tag{3.8}$$

It follows that

$$\begin{aligned} if\ s = nKS_1,\ then\ t_1 &= KnS_1 - \theta_1(nKS_1) \\ &= KnS_1 - \delta_{1n} \qquad (3.9) \\ if\ s = nKS_2,\ then\ t_2 &= KnS_2 - \theta_2(nKS_2) \\ &= KnS_2 - \delta_{2n} \qquad (3.10) \end{aligned}$$

Therefore, the transformations in eq.(3.7) and eq.(3.8) transform the points nKS_1 and nKS_2 of the s-axes into the points $KnS_1 - \delta_{1n}$ and $KnS_2 - \delta_{2n}$ of the t-axes. Considering that the sampling with respect to the event, s, has the same period for all the signals, that is $S_1 = S_2$. Assuming that V_e is generated instantaneously once F_e is rendered, that is $\delta_{1n} = \delta_{2n}$. Then, $KnS_1 - \delta_{1n} = KnS_2 - \delta_{2n}$ and the sampling instances of signals $F_h(t)$ and $V_h(t)$ are identical. A similar analysis can be made to show that the sampling instances of signals $F_e(t)$ and $V_e(t)$ are identical. For the time-based control case, since the time and event sampling periods are not the same for either $F_h(t)$ and $V_h(t)$ or $F_e(t)$ and $V_e(t)$, then similar conclusions cannot be made.

This conclusion implies that, in the event-based control case if the system is event-transparent then,

$$F_e(s)_{rd} = F_e(s)_{nd} \Rightarrow F_h(s)_{rd} = F_h(s)_{nd} \tag{3.11}$$

Assuming that the operator's reaction is consistent based on the force felt, then

eq.(3.11) gives

$$V_h(s)_{rd} = V_h(s)_{nd} \Rightarrow V_e(s)_{rd} = V_e(s)_{nd} \tag{3.12}$$

$\Diamond$

Thus the system's control commands are the same regardless of random delay as stated previously, implying that its performance with respect to the event-based action reference is transparent to delay and its randomness. Since the event-based action reference can depend on any system output, which is a non-decreasing function of time [6], this property implies that consistent system behavior with respect to a certain system output can be achieved. However, special care should be taken in designing the event-based action reference. It has to ensure that $F_e(s)$ is the same regardless of time delay. For example, $F_e(s)$ can be a function of the distance the robot moved; therefore, the event-based reference, s, can be chosen to be the distance traveled.

The relationship between event-transparency and time-transparency for event-based control systems can be described by the following theorem:

Theorem 3 *An event-transparent system under no delay conditions preserves time-transparency.*

Proof. To start, the signals $F_e(t)$ and $F_h(t)$ are examined under no delay conditions. Assume that the deviations of the nonuniform samples of these signals from uniform samples are δ_n. These deviations are a function of the system output and thus the commands, $V_e(s)$. In the general case, δ_n will also be a function of time delay. However this analysis is done for the no delay case; therefore, δ_n are only a function of the system output and they are the same for both signals. Assuming that

the sampling period with respect to the event-based reference is S define the function

$$\theta(s) = \sum_{n=-\infty}^{\infty} \delta_n \frac{\sin w_1(s - nKS)}{w_1(s - nKS)} \tag{3.13}$$

Sample values of $\theta(s)$ are δ_n,

$$\theta(nKS) = \delta_n \tag{3.14}$$

The transformation between the s-axis and t-axis is

$$t = s - \theta(s) \tag{3.15}$$

It follows that

$$\begin{aligned} if\ s = nKS,\ then\ t &= KnS - \theta(nKS) \\ &= KnS - \delta_n \end{aligned} \tag{3.16}$$

Since for all m and n, if $s_m < s_n \Rightarrow t_m < t_n$ and since two events can not occur at the same time instant then eq.(3.15) is a strictly increasing function of s and its inverse exists:

$$s = \gamma(t) \tag{3.17}$$

Define the functions

$$g_h(s) \equiv F_h[s - \theta(s)] = F_h(t) \tag{3.18}$$

$$g_e(s) \equiv F_e[s - \theta(s)] = F_e(t) \tag{3.19}$$

$$\Rightarrow g_h(nKS) = F_h(nKS - \delta_n) \tag{3.20}$$

$$and \quad g_e(nKS) = F_e(nKS - \delta_n) \tag{3.21}$$

Hence, $g_h(s)$ and $g_e(s)$ are known at a sequence of equidistant points $s = nKS$. If these signals are band limited by w_1 then S can be chosen such that Shannon's sampling condition is satisfied:

$$\frac{\pi}{KS} \geq w_1 \tag{3.22}$$

and then the following equations will hold:

$$g_h(s) = \sum_{n=-\infty}^{\infty} F_h(nKS - \delta_n) \frac{\sin w_1(s - nKS)}{w_1(s - nKS)} \tag{3.23}$$

$$g_e(s) = \sum_{n=-\infty}^{\infty} F_e(nKS - \delta_n) \frac{\sin w_1(s - nKS)}{w_1(s - nKS)} \tag{3.24}$$

and therefore,

$$\begin{aligned} F_h(t) &= g_h[\gamma(t)] \qquad (3.25) \\ &= \sum_{n=-\infty}^{\infty} F_h(nKS - \delta_n) \frac{\sin w_1(\gamma(t) - nKS)}{w_1(\gamma(t) - nKS)} \\ F_e(t) &= g_e[\gamma(t)] \qquad (3.26) \\ &= \sum_{n=-\infty}^{\infty} F_e(nKS - \delta_n) \frac{\sin w_1(\gamma(t) - nKS)}{w_1(\gamma(t) - nKS)} \end{aligned}$$

From eq.(3.3) it implies that

$$F_h(nKS - \delta_n) = \alpha F_e(nKS - \delta_n) \tag{3.27}$$

which implies that for the no delay case, an event-based control system results in

$$F_h(t) = \alpha F_e(t) \tag{3.28}$$

Similarly,

$$V_h(t) = \beta V_e(t) \tag{3.29}$$

which results in

$$\Rightarrow \frac{F_h(t)}{V_h(t)} = C\frac{F_e(t)}{V_e(t)} \tag{3.30}$$

Therefore, an event-transparent system under no delay conditions preserves time-transparency.

$\Diamond$

However this is not true for the random delay case because the δ_n would be a function of delay. Thus eq.(3.27) would not hold because F_h and F_e are not deformed similarly. Note that all this analysis is done independent of what F and V are, which implies that this analysis applies to any kind of feedback and control commands used. So event-based control systems, which are event-transparent, are able to preserve time-transparency under no delay and are consistent in their control. On the other hand, time controlled systems are not able to ensure any consistency when faced with random delay.

3.4 Event-Synchronization Analysis

As for synchronization, research found in the literature has mainly focused on time synchronization in open loop systems [15]-[20]. However, the control systems pre-

sented in this research are closed loop event-based. Therefore, there is a need to define synchronization in this context. Event-synchronization is defined as:

Definition 2 *An event-synchronized system is one in which all the signals (control and feedback) in the system are always referencing the same event-based reference.*

This definition is similar to the definition of time synchronization but instead of having the time as a reference the event-based action reference is being used. Also an important difference is that this definition includes the control signal too, which means that the control has also to be synchronized with the feedback. This implies that the event-synchronized system ensures that the feedback sensed is a reflection of the system's most current state.

From the definition it implies that systems have to be designed in a way such that all the supermedia streams being rendered have the same action reference stamps. However, this requirement might not be feasible just like perfect time-synchronization of video and audio transmitted over the Internet is not. Therefore, in some cases; such as, force and video synchronization, a certain tolerance should be permitted. Typically, a master supermedia stream would be chosen and synchronization would be carried out with respect to it. For example, with force and video, force can be the master stream since it is more informative and requires less bandwidth. In this case a video frame will be displayed if its event-based stamp is within a certain predefined tolerance range from the force sample being rendered; otherwise, it is discarded.

In addition, event-synchronized systems have to ensure that the control signal has the same action reference stamp as the feedback being rendered. Examining the event-based case in Figure 3.2, it is clear that the update of signals in both directions is not triggered by time. Also since the buffering effect of delay is eliminated then the haptic force felt by the operator is the most up-to-date one; there could not have been any change in the system state meanwhile because there are no commands flowing in the network. The same thing applies to the command received by the robot, it is

the most up-to-date one since there were no new haptics flowing in the network that could have generated new commands. This implies that the operator and the robot are always synchronized in event regardless of time delay and its variation. In the case of multiple remote operators and/or multiple robots, event-synchronization also ensures that the operators are synchronized with each other and with the different robots.

A fundamental difference between time-based synchronization and event synchronization is that the latter results in content or state-based synchronization. When synchronization is done based on the event stamp, which is a reflection of the state of the remote robot, it implies that the supermedia streams being rendered where sensed at relatively close system states. As for time-based synchronization, it results in synchronization with respect to time regardless of the system state. To compare, consider the case of a slowly operating system. In the time-based case, if a frame is slightly "old" it will be discarded although the state of the system did not change significantly; however, in the event-based case the frame will still be displayed since the event or state of the system did not change considerably. On the other hand, in the case of fast operation and movement, a relatively "new" frame will be rendered in the time-based case although the state of the system has changed drastically. The same frame will be discarded in the event-based case since what it shows reflects an "old" state of the system because of the quick changes the system is going through. Therefore, event-synchronization is done based on the content or information being rendered; where as, time-based synchronization is done based on when and not what is sensed.

The next chapter covers the modeling of event-based real-time teleoperation systems. The dynamic models of several general systems are given. In addition, Petri Net, as a modeling and analysis tool, is proposed for teleoperation systems. At the end a comparison with a popular modeling and analysis approach is included.

CHAPTER 4

MODELING OF EVENT-BASED REAL-TIME TELEOPERATION SYSTEMS

This chapter details two modeling techniques used for event-based real-time teleoperation systems. Two cases of the dynamic model are presented; the first one is of a general teleoperation system with a single operator and single robot, the second one is of a general tele-cooperation system with multi-operators at multi-site controlling multi-robots. These two cases are also modeled using a new proposed modeling technique for event-based real-time teleoperation systems. This technique uses Petri Nets as a modeling tool because of its many attractive features, which will be discussed. In addition, a comparison is made between the models suggested in this research and other models used in the literature, concentrating on the one presented in [4].

4.1 Modeling and Analysis

In this section the dynamic models of two general event-based real-time teleoperation systems are given. The first is a model of a teleoperation system with a single operator and single robot. The second one is a model of a general tele-cooperation system with multi-operators at multi-site controlling multi-robots.

4.1.1 Teleoperation System: Single Operator and Single Robot

The detailed model of a general teleoperation system with force feedback is discussed in this section. Figure 4.1 gives a general block diagram of a traditional teleoperation system and Figure 4.2 presents a general schematic diagram of some of the systems developed in this research. The modeling of the blocks in Figure 4.1 and Figure 4.2 will be discussed in details and each term is explained in Table 4.1.

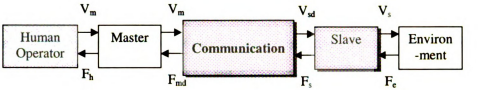

Figure 4.1: Block diagram of a traditional teleoperation system.

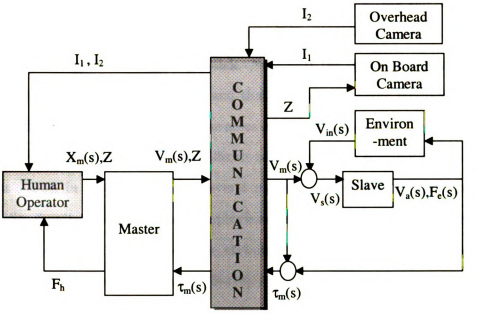

Figure 4.2: Detailed block diagram of some of the event-based teleoperation systems developed.

Block	Traditional Variables	Our Variables
Human Operator	F_h: applied force	$F_h \in \Re^3$: applied force $X_m \in \Re^3$: joystick position
Master	V_m: velocity desired	$V_m \in \Re^3$: velocity desired
Communication Link	F_{md}: desired force V_{sd}: desired velocity	
Slave	V_s: velocity F_s: force	$V_a \in \Re^3$: actual velocity $\tau_m \in \Re^3$: desired force
Environment	F_e: contact force	$V_{in} \in \Re^3$: virtual/real contact effect $F_e \in \Re^3$: contact force $V_s \in \Re^3$: velocity set

Table 4.1: The explanation of the various variables in the teleoperation systems.

As Figure 4.1 shows, the general teleoperation system consists of a human operator, master (usually the master is a joystick), communication network, slave (usually the slave is a robot.), and the environment. The same applies to the systems developed as part of this research as shown in Figure 4.2 but with some modifications mostly in the connections.

In what follows, each block in the system will be modeled giving the details of each variable and the equations that relate them. The case considered is velocity control using a 3 degrees of freedom joystick. Note that the dynamic equations are in terms of the event-based reference, s, and that the robot is being controlled with 3 degrees of freedom, which makes the variables 3 term vectors.

Human Operator: This is the most difficult to model, but a spring-like behavior, which was shown in [73] and used in several places in the literature [74], can be assumed. So, despite all the internal complexities, the wrist has a spring-like effect. In addition, the human compensates for certain machine instabilities making the coupled human-machine system stable [75]-[78].

As force is felt, the operator will generate a new joystick position according to the following:

$$X_m(s+1) = \frac{F_h(s)}{K_h} \tag{4.1}$$

where K_h is a constant and s is the event-based reference, $s \in \{1, 2, \cdots\}$. X_m and F_h are

$$X_m(s) = \begin{bmatrix} X_{mx}(s) \\ X_{my}(s) \\ X_{m\theta}(s) \end{bmatrix} \quad F_h(s) = \begin{bmatrix} F_{hx}(s) \\ F_{hy}(s) \\ F_{h\theta}(s) \end{bmatrix} \tag{4.2}$$

As Table 4.1 indicates $F_h(s)$ is the applied force, which means the force that the

operator feels. Thus, the x and y components are due to the force fed back and to the additional force required to get the joystick to the new location. Since force is not fed back in the θ direction, this component is just a result of getting the joystick to the next location. As seen in eq.(4.1), $X_m(s+1)$ is related to $F_h(s)$, so $X_m(s)$ at reference $s+1$ is generated by the previous force at reference s. This results in an event-based system where each event is triggered by the previous one.

Master (Joystick): The dynamics of the joystick are

$$M_m \dot{V}_{mm}(s) = F_h(s) + \tau_m(s) \tag{4.3}$$

where M_m is the mass of the joystick handle, V_{mm} is velocity of joystick movement, F_h is as described before and τ_m is the feedback from the robot, which would be the force rendered by the joystick. The result from these dynamics is the joystick getting to a new position $X_m(s+1)$. From this position the desired velocity V_m is derived according to

$$V_m(s) = K_m X_m(s) \tag{4.4}$$

where K_m is a scaling constant, $X_m(s)$ is as before and $V_m(s)$ is the desired velocity of the robot. The different vectors are

$$V_m(s) = \begin{bmatrix} V_{mx}(s) \\ V_{my}(s) \\ V_{m\theta}(s) \end{bmatrix} \quad \tau_m(s) = \begin{bmatrix} \tau_{mx}(s) \\ \tau_{my}(s) \\ 0 \end{bmatrix} \tag{4.5}$$

Eq.(4.5) shows that $\tau_{m\theta}(s) = 0$, since force is not fed back in the rotational direction.

Communication Block (Internet): Resulting from event-based control, the communication link is simply a delay element that plays no role in the modeling of the

system. It is assumed that the Internet is simply supplying the link and in case this connection is lost, the system will simply stop awaiting the connection. Since the advance of time does not affect the system and only the advance of the event-based reference s will, when the connection is lost the system will remain stable and will resume action after the connection is re-established. This makes the system very robust since no initialization or resynchronization is required.

Environment: Interaction with the environment can be modeled in two ways depending on the system being considered. The first case is the one involving the control of a mobile robot and the second case is the one involving either a mobile manipulator, a manipulator or a micro controller. For the first case, the force is a virtual force which is based on the tracking error of the robot. Whereas in the second case, the force corresponds to actual physical force detected.

For the first case, contact is not required to generate force. Instead, different sensors around the robot are used to detect obstacles. Based on the distance between the obstacles and the robot the velocity is reduced, by that creating a tracking error, which is considered as a *virtual force*. This reduction in the velocity is done according to a function of the distance $f(d)$, which gives a velocity $V_{in}(s)$ to subtract from the desired velocity $V_m(s)$. So the velocity set becomes

$$V_s(s) = V_m(s) - V_{in}(s) \tag{4.6}$$

where $V_s(s)$ is the velocity set to the robot, $V_{in}(s)$ is the effect of the environment and $V_m(s)$ is as before. Clearly, the desired velocity that the robot receives is less than the actual one, this slowing down is used to generate the virtual force to be fed back. Note that no force is generated in the rotational direction that is why $V_{in\theta}(s) = 0$ and $V_{s\theta}(s) = V_{m\theta}(s)$, as seen in eq.(4.7).

$$V_s(s) = \begin{bmatrix} V_{sx}(s) \\ V_{sy}(s) \\ V_{m\theta}(s) \end{bmatrix} \quad V_{in}(s) = \begin{bmatrix} V_{inx}(s) \\ V_{iny}(s) \\ 0 \end{bmatrix} \tag{4.7}$$

For the second case, involving either a mobile manipulator, a manipulator or a micro controller, the force fed back corresponds to physical force detected by the force/torque sensor. So the force is generated when actual contact is established with the environment.

Slave (Mobile robot): The first case of a slave is the mobile robot. The robot receives $V_s(s)$, it will be commanded to move with that velocity but would actually have $V_a(s)$ as its velocity. The robot will then calculate the velocity tracking error with respect to the original desired velocity $V_m(s)$ and send that back to the master to be rendered as the virtual force due to the environment. So $\tau_m(s)$ and the dynamics of the robot are

$$\tau_m(s) = K_\tau(V_m(s) - V_a(s)) \tag{4.8}$$

$$M_s\dot{V}_a(s) = F_e(s) + \tau_s(s) \tag{4.9}$$

$$\tau_s(s) = -\gamma V_a(s) + KV_{err}(s) - \alpha_f F_e(s)$$

$$V_{err}(s) = V_s(s) - V_a(s) \tag{4.10}$$

$$V_a(s) = \begin{bmatrix} V_{ax}(s) \\ V_{ay}(s) \\ V_{m\theta}(s) \end{bmatrix} \tag{4.11}$$

where K_τ, γ, K and α are constants, M_s is mass of robot, F_e is the actual environment forces if any and usually assumed very small. τ_s and V_{err} are robot internal

controller terms. Eq.(4.8) shows that what the operator is feeling is actually the velocity tracking error, which could be a result of the robot getting close to an object. This implies that from the direction and magnitude of the force, the operator feels the location and the distance of the obstacle to the robot. The important point to note is that the operator will still feel force whenever the robot comes in contact with an object. Actual contact that results in a decrease in $V_a(s)$ will result in a proportional increase in $\tau_m(s)$. Also, $V_{a\theta}(s) = V_{m\theta}(s)$ since $\tau_{m\theta} = 0$.

Slave (manipulator): The second case of a slave is a manipulator arm. The dynamic model for the robot arm can be written as

$$D(q)\ddot{q} + c(q, \dot{q}) + g(q) = u \tag{4.12}$$

where q is the 6×1 vector of joint displacements, $\dot{q}$ is the 6×1 vector of joint velocities, u is the 6×1 vector of applied torques, $D(q)$ is the 6×6 positive definite manipulator inertia matrix, $c(q, \dot{q})$ is the 6×1 centripetal and coriolis torques, and $g(q)$ is the 6×1 vector of gravity term.

Let $y \in \Re^6$ be a task space vector defined by $Y = (x, y, z, O, A, T)^T$. $(x, y, z)^T$ denotes the position of the end-effector in the cartesian space, $(O, A, T)^T$ denotes an orientation representation (Orientation, Altitude and Tool angles). After applying non-linear feedback with $u = D(q)J^{-1}(-\dot{J}\dot{q} + v) + C(q, \dot{q}) + g(q)$ [79]-[82]. The dynamic model of the arm can be linearized to $\ddot{Y} = v$.

In this case the force fed back, $\tau_m(s)$, would be

$$\tau_m(s) = K_\tau F_e \tag{4.13}$$

where K_τ is a constant and F_e is the physical contact force detected.

Slave (mobile manipulator): The mobile manipulator is composed of a mobile robot and a robot arm. However, the two subsystems cannot be modeled separately,

force interaction between the mobile platform and the robot arm should be considered. The model of a mobile manipulator can be described as

$$\begin{bmatrix} M_1\bar{J}^{-1} & N_1 \\ N_2 & M_2 \end{bmatrix} \cdot \begin{bmatrix} \ddot{x}_1 \\ \ddot{x}_2 \end{bmatrix} + \begin{bmatrix} -M_1\bar{J}^{-1}\dot{\bar{J}}\dot{q}' + c_1 + g_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{4.14}$$

where $x_1 = \{x, y, z, O, A, T\}^T, x_2 = \{x_b, y_b, \theta_b\}^T$. Here x_1 denotes the position and orientation of the end-effector of the mobile manipulator, x_2 denotes the position and orientation of the mobile platform. M_1, M_2 are the inertia matrices and the N_1, N_2 describe the interaction between each other. τ_1, τ_2 are generalized input torques for the robot arm and mobile platform respectively. Defining $X = \{x, y, z, O, A, T, x_b, y_b, \theta_b\}^T$ and applying nonlinear feedback to the system described in eq.(4.14), the system can be decoupled and linearized as

$$\ddot{X} = v$$

where v is internal control input vector. It can be easily seen from the model that the motion is redundant. The robot arm posture will be determined by $x - x_b$ and $y - y_b$. The operator will give command to change x and y, the motion of the mobile platform will be determined by the current posture of the robot arm. If the robot arm is almost fully extended or extracted, the mobile platform will be repositioned to achieve a better posture. In this case the force fed back, $\tau_m(s)$, would be

$$\tau_m(s) = K_\tau F_e \tag{4.15}$$

where K_τ is a constant and F_e is the physical contact force detected.

Cameras: Two cameras are used to supply video feedback. The one on-board the robot receives pan, tilt and zoom commands Z from the operator and sends back the video image I_1. The other camera is overhead and it sends the second image, I_2, to the operator.

4.1.2 Tele-cooperation System: Multi Operators and Multi Robots

The general structure of a multi-operator multi-robot collaborative teleoperation with force reflection system is shown in Figure 4.3 and all the terms are explained in Table 4.2. This model has many similarities with the one discussed in the previous section; however, the force is generated using a new concept. Figure 4.3 shows that the force fed back to each operator can correspond to the other operator's desired velocities, the robots' interaction with the environment or even a combination of both. The designer choice of a particular force scenario depends on the particular task setup.

The special case presented in this research, where a mobile manipulator was used, is modeled in Figure 4.4. Each block will be discussed in detail and all terms are explained in Table 4.3.

Human Operator: As discussed previously, a spring-like behavior may be assumed for the operators. Once the operators feel a force, they will generate a new joystick position according to the following:

$$X_m(s+1) = \frac{F_p(s)}{K_m} \qquad X_p(s+1) = \frac{F_m(s)}{K_p} \tag{4.16}$$

where K_m and K_p are scaling constants and s is the event-based reference, where $s \in \aleph$. X_m, X_p, F_m and F_p are

$$X_m(s) = \begin{bmatrix} X_{mx}(s) \\ X_{my}(s) \\ X_{m\theta}(s) \end{bmatrix} \quad X_p(s) = \begin{bmatrix} X_{px}(s) \\ X_{py}(s) \\ X_{p\theta}(s) \end{bmatrix} \tag{4.17}$$

$$F_m(s) = \begin{bmatrix} F_{mx}(s) \\ F_{my}(s) \\ F_{m\theta}(s) \end{bmatrix} \quad F_p(s) = \begin{bmatrix} F_{px}(s) \\ F_{py}(s) \\ F_{p\theta}(s) \end{bmatrix} \tag{4.18}$$

As Table 4.2 shows, $F_m(s)$ and $F_p(s)$ are the applied forces, that is the forces

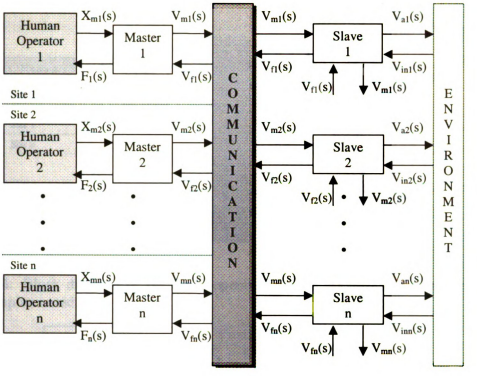

Figure 4.3: Detailed block diagram of a general multi-operator multi-robot tele-collaborative control system.

Block	Our Variables
Human Operator	$F_i \in \Re^3$: Applied force $X_{mi} \in \Re^3$: Joystick position
Master	$V_{mi} \in \Re^3$: Velocity desired
Slave	$V_{ai} \in \Re^3$: Actual velocity $V_{fi} \in \Re^3$: Feedback
Environment	$V_{ini} \in \Re^3$: Real or Virtual contact

Table 4.2: Explanation of the various variables in Figure 4.3.

that the operators feel. As seen in eq.(4.16), $X_m(s+1)$ and $X_p(s+1)$ are related to $F_p(s)$ and $F_m(s)$. So $X_m(s+1)$ and $X_p(s+1)$ at reference $s+1$ are generated by the previous force at event s, as discussed previously.

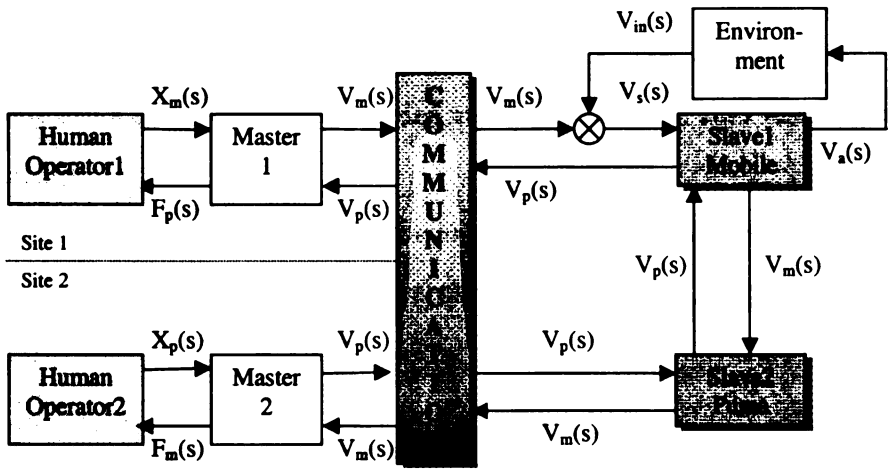

Figure 4.4: Block diagram of the multi-operator mobile manipulator teleoperation system implemented.

Block	Our Variables
Human Operator	$F_m, F_p \in \Re^3$: Applied force $X_m, X_p \in \Re^3$: Joystick position
Master	$V_m, V_p \in \Re^3$: Velocity desired
Communication Link	
Slave	$V_a \in \Re^3$: Actual velocity
Environment	$V_{in} \in \Re^3$: Virtual contact $V_s \in \Re^3$: Velocity set

Table 4.3: Explanation of the various variables in Figure 4.4.

Master (Joystick): The dynamics of the joysticks are

$$M_m \dot{V}_{mm}(s) = F_p(s) + F_{V_p}(s) \tag{4.19}$$

$$M_p \dot{V}_{mp}(s) = F_m(s) + F_{V_m}(s) \tag{4.20}$$

$$F_{V_m}(s) = C_m V_m(s) \tag{4.21}$$

$$F_{V_p}(s) = C_p V_p(s) \tag{4.22}$$

where M_m and M_p are the masses of the joysticks' handles, V_{mm} and V_{mp} are the velocities of the joysticks' movement, and F_m and F_p are as described earlier. F_{V_m} and F_{V_p} are the forces played by the joystick, which are simply the velocities V_m and V_p

fed back from the robots scaled by the constants C_m and C_p respectively. This implies that the operators feel each others intended velocities, this way they can collaborate more efficiently.

The result of these dynamics is that the joysticks move to new positions $X_m(s+1)$ and $X_p(s+1)$. From these positions the desired velocities V_m and V_p are derived according to

$$V_m(s) = K_{m1}X_m(s) \qquad V_p(s) = K_{p1}X_p(s) \tag{4.23}$$

where K_{m1} and K_{p1} are scaling constants, $X_m(s)$ and $X_p(s)$ are as before. $V_m(s)$ and $V_p(s)$ are the desired velocities of the mobile and the puma respectively. The velocity vectors are

$$V_m(s) = \begin{bmatrix} V_{mx}(s) \\ V_{my}(s) \\ V_{m\theta}(s) \end{bmatrix} \quad V_p(s) = \begin{bmatrix} V_{px}(s) \\ V_{py}(s) \\ V_{p\theta}(s) \end{bmatrix} \tag{4.24}$$

Communication Block (Internet): The same discussion that was given in the previous section applies here.

Environment: Sensors on the mobile robot are used to detect objects. Based on the distance between the object and the robot, velocity is reduced. This is calculated according to a function of the distance from the object, $0 \leq f(d) \leq 1$, that is multiplied by $V_m(s)$ to give a velocity value $V_{in}(s)$. $V_{in}(s)$ is subtracted from the desired velocity $V_m(s)$ to give the velocity set for the robot $V_s(s)$:

$$V_s(s) = V_m(s) - V_{in}(s) \tag{4.25}$$

As a result, the robot gets a velocity from the server that is less than the one desired by the operator.

$$V_s(s) = \begin{bmatrix} V_{sx}(s) \\ V_{sy}(s) \\ V_{m\theta}(s) \end{bmatrix} \quad V_{in}(s) = \begin{bmatrix} V_{inx}(s) \\ V_{iny}(s) \\ 0 \end{bmatrix} \tag{4.26}$$

Slave1 (Mobile robot): The dynamic equations modeling the mobile were presented in the previous section.

Slave2 (PUMA manipulator): The dynamic equations modeling the manipulator were presented in the previous section.

4.2 Petri Net Model

One of the difficulties of designing and implementing real-time teleoperation systems is the lack of a general modeling technique. There is a need for a model that can capture the concurrence, non-determinism and logical behavior of such systems. This model should be easily analyzed to study the underlying system performance. Properties such as stability and synchronization cannot be easily studied using the current models. For this purpose, Petri Nets are suggested as an efficient way to model the concurrence and complexity of Internet based teleoperation systems [83]-[90].

Petri Nets have several attractive features, which are based on their ability to describe and study systems that are characterized as being concurrent, asynchronous, distributed, parallel, non-deterministic and/or stochastic. Thus, it is a very adequate and efficient tool for studying Internet based telerobotic systems. Although Petri Nets have been used before for modeling and studying robotic systems, that was limited to manufacturing and scheduling related problems [84]. In addition, based on this modeling technique, different properties of the system, some of which can be linked to physical properties, can be studied using Petri Nets analysis tools. Another attractive feature of Petri Nets is the ability to setup state equations, algebraic equations and

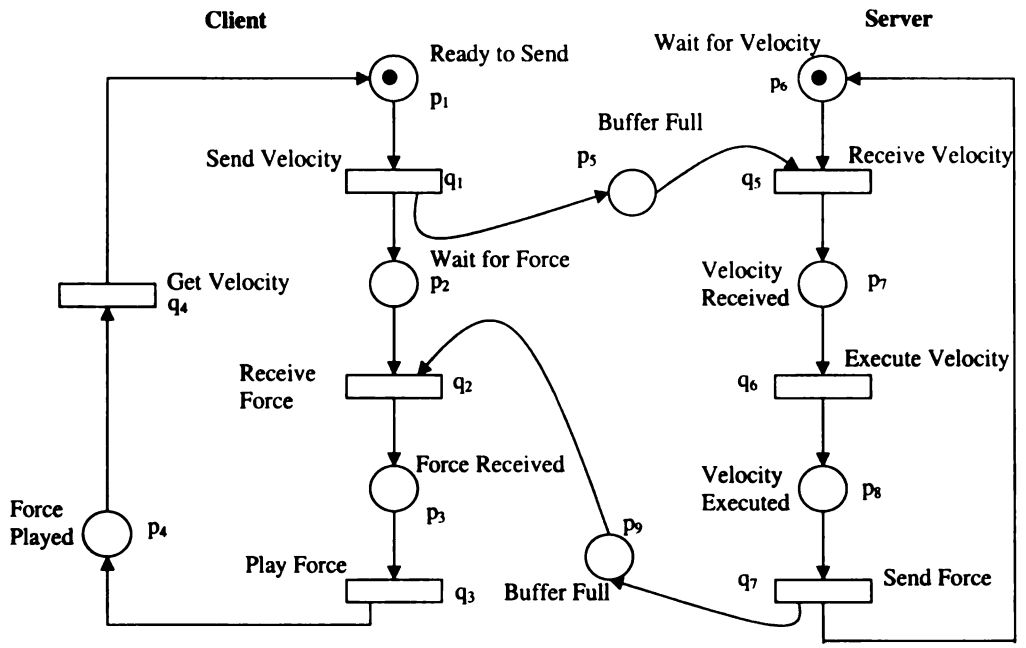

Figure 4.5: Petri Net model of the system shown in Figure 4.2.

other mathematical models governing the behavior of the system [83].

To take advantage of these feature, Petri Nets are used to model event-based teleoperation systems. The general teleoperation system shown in Figure 4.2 is modeled in Figure 4.5 using a Petri Net, which is a directed graph that has an initial state (marking) [83]. As seen, this graph consists of two kinds of nodes: places and transitions, where arcs are either from a place to a transition or vice versa. Places represent conditions and system state, such as being "ready to send". Transitions represent events and actions, such as "play force". Tokens, which are represented by dots in places, correspond to signals flowing in the system and their collection reflects the state of the system.

To capture the dynamic behavior of the system, the state of the system is changed according to the following transition (firing) rule [83]:

-A transition, such as "send velocity", is enabled if each input place, in this case "ready to send", has a token in it.

-An enabled transition may or may not fire (depending on whether or not the

event actually takes place).

-A firing of an enabled transition, such as "send force", removes a token from each input place, in this case "velocity executed", and adds a token to each output place, in this case "buffer full" and "wait for velocity".

This description clearly implies that Petri Nets are an adequate tool to model event-based systems, such as the one presented in [70]. Since as in event-based systems each firing is enabling the next one and no event can occur out of order. As for the dynamics of different system components, they are represented by transitions. For example, "play force" in the mobile client represents $M_m \dot{V}_{mm}(t) = F_p(t) + F_{V_p}(t)$.

Another strength of Petri Nets is their support for analysis of many important properties associated with the systems they model. These properties are divided into properties that are dependent on the initial marking (behavioral properties) and these properties which are independent of the initial marking (structural properties) [83] [85]. In this study interest is in the following behavioral properties:

Boundedness: The model presented is k-bounded or simply bounded if the number of tokens in each place does not exceed a finite number k for any marking reachable from the initial marking. It is said to be safe if it is 1-bounded [83]. This implies that there is no accumulation of tokens in places at any time implying that the system is stable [87].

Liveness: The model is said to be live if, no matter what marking has been reached from the initial state, it is possible to ultimately fire any transition of the net by progressing through some further firing sequence. This property guarantees deadlock-free operation [83]. This implies that, despite the non-determinism in the system, there is no case that would cause the system to stop operating normally.

Several analysis methods exist which may be classified into the three groups: Coverability (reachability) tree method, Matrix-equation approach, Reduction or decomposition techniques [83]. The first method involves the enumeration of all reachable

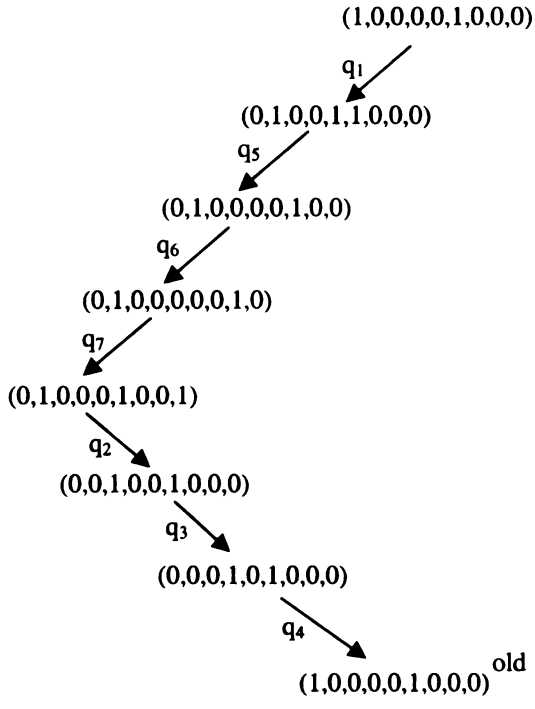

Figure 4.6: The coverability tree of the model shown in Figure 4.5.

markings or their coverable markings. It can be applied to all classes of nets, but is limited to small nets due to the complexity of the state-space explosion. The other two techniques are powerful but they can only be applied to special subclasses of Petri Nets.

The general teleoperation system coverability tree is shown in Figure 4.6. From this tree it can be deduced that the system is bounded and live. Boundedness is deduced from the fact that all the nodes in the tree have ones and zeros in them. This implies that not only the system is bounded but also safe [83]. Which, in other words, means that the number of tokens in any place does not exceed one.

In addition, the coverability tree conveys that the system is dead-lock free since all the markings in the tree have enabled transitions [83]. So the system is both bounded (safe) and live. Since the system is safe it is clear that each place in the system can have only one token at most at any point in time. And since the passage of tokens

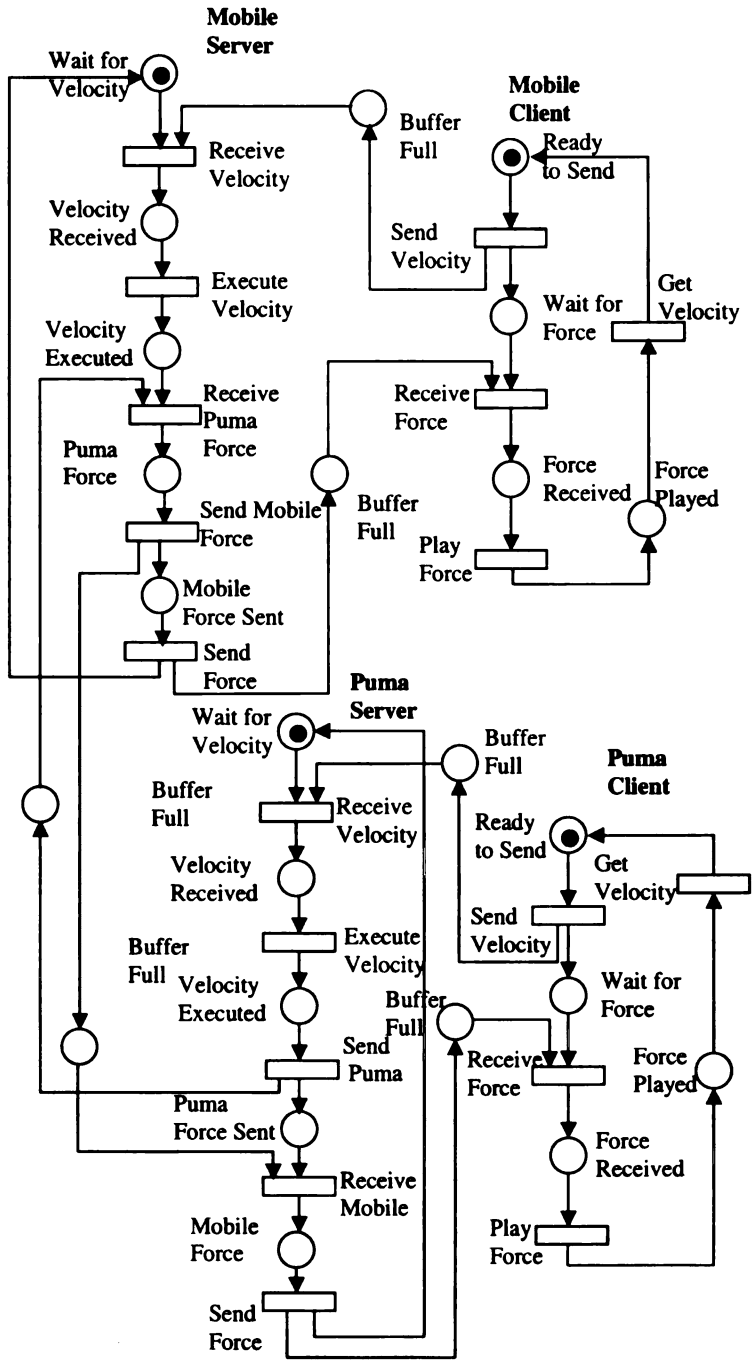

Figure 4.7: Petri Net model of the system shown in Figure 4.4.

in a place represents the advance of the event-based reference, s, for that place and since it is clear from the model that a place will not receive another token until all the

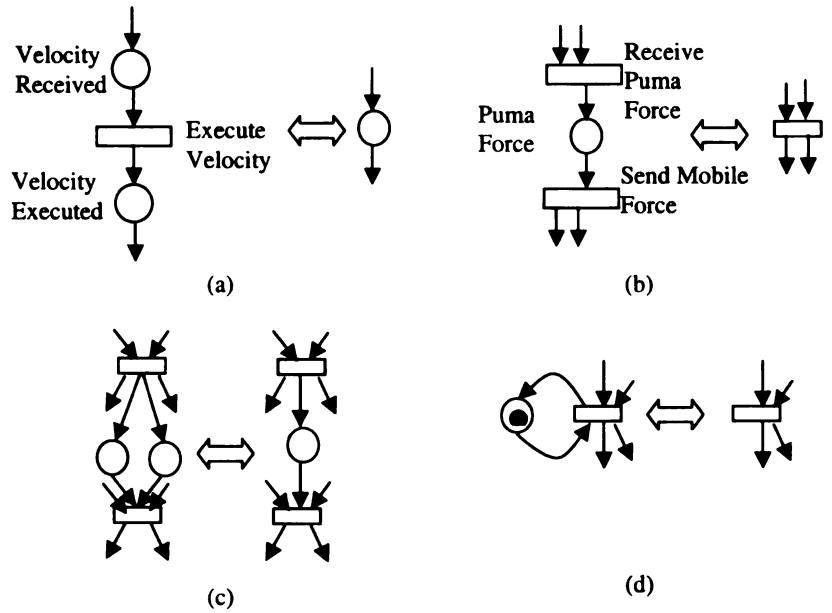

Figure 4.8: Reduction rules used to transform the model into a simpler one (a) Fusion of series places (b) Fusion of series transitions (c)Fusion of parallel places (d) Elimination of self-loop places.

other places have received a token then the system is event-synchronized. This will be formally stated and a detailed design procedure will be given in the next chapter. It is worth noting that in systems where delays are in the order of seconds synchronization becomes as important an issue as stability. Large random time delays would make any form of feedback, whether video or force, worthless if the different system parts are not synchronized. Since receiving feedback that does not reflect the most up-to-date state of the robot would result in the wrong control commands being sent. But the existence of random time delay makes time synchronization impossible to achieve. That is why event-synchronization was presented in [70]. Event-synchronization ensures that the feedback received corresponds to the most up-to-date state of the system and thus the right control signal would be sent.

As for the tele-cooperation system with multi-operators and multi-robots, which is shown in Figure 4.4, it can be modeled using Petri Nets as seen in Figure 4.7.

To facilitate the analysis of the Internet based multi-operator at multi-site collab-

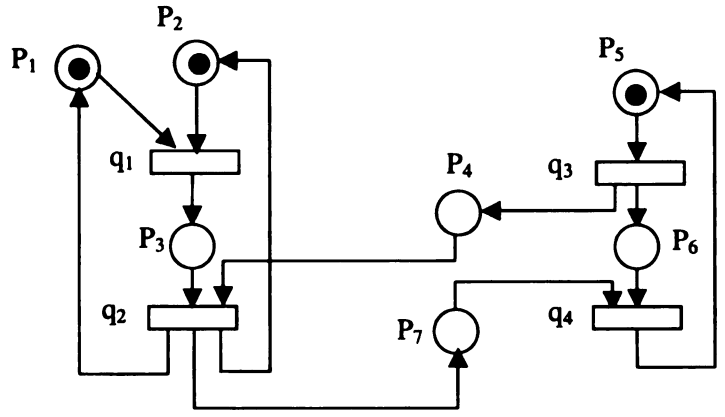

Figure 4.9: The reduced form of the model shown in Figure 4.7.

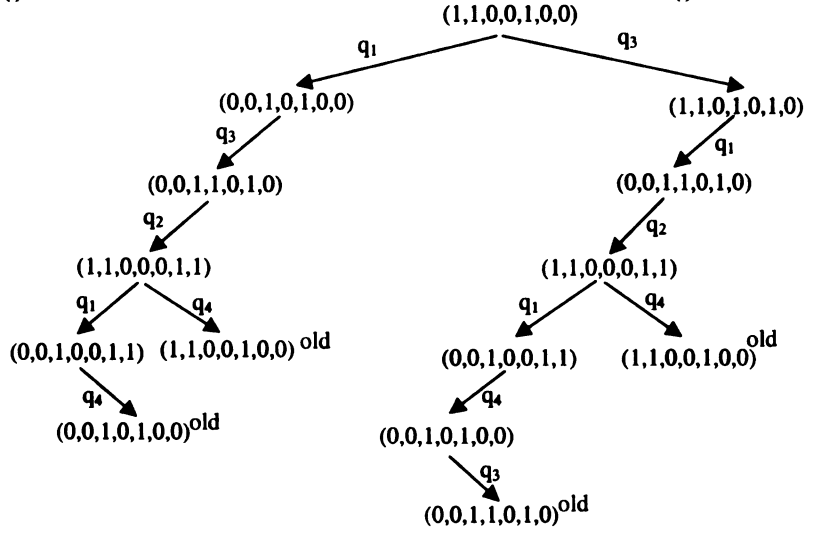

Figure 4.10: The coverability tree of the reduced model.

orative teleoperation system, while preserving properties of boundedness and liveness, the system model is reduced to the one shown in Figure 4.9 [83]. The reduction rules used for this purpose are shown in Figure 4.8. It is worth mentioning that the model could have been further simplified but the one presented is simple enough to develop a reasonable size coverability tree.

The reduced system's coverability tree is shown in Figure 4.10. From this tree it can be deduced that the system is bounded and live. Moreover, this coverability tree conveys that the system is dead-lock free since all the markings in the tree have

enabled transitions [83]. Also, following a similar analysis to the one done for the general teleoperation system case, it can be deduced that this system is also event-synchronized. This link between Petri Net properties and event-synchronization is formally developed based on the definition given earlier as,

Definition 3 *An event-synchronized system is one in which all the signals in the system are always referencing the same event-based reference.*

For the control and the feedback signals no tolerance is allowed; therefore, they should have the same event-based reference at any instant. This is required so that the feedback being rendered is guaranteed to be a reflection of the most up-to-date state of the system. This definition translates into two requirements; first no two instances of the same signal can coexist anywhere in the network and second every feedback instant or sample should generate one and only one command response and vice versa.

These two event-synchronization requirements can be translated into Petri Nets properties. First there is a need to ensure that the Petri Net will not dead-lock, which formally means the network has to be shown as "live" [83]. Second, it is required not to have two instances of the same signal at any point, that is no accumulation of signals at any point within the system. This translates into having either one or no tokens in any place in the net. For this to be true the Petri Net has to be "safe" [83]. This however does not guarantee not having instances of the same signal in other places in the network. For this to be true and to satisfy the second condition of event-synchronization the Petri Net model has to be "fair" with a reproduction vector x, where $x = [1, 1, ..., 1]$ [91]. A fair network with such a reproduction vector guarantees that all transitions will fire equally many times, which implies that once a transition fires it cannot fire again until all the transitions in the network have fired. So the system executes each step once and only once in each communication cycle, which implies that one and only one control signal will be generated for each feedback

sample and vice versa. Formally these conditions and their implications can be stated as,

Theorem 4 *A system is event-synchronous if and only if the Petri Net describing the system's communication protocol is live, safe and fair with a reproduction vector* x_{nx1}, *where* $x = [1, 1, ...1]$ *and* n *is the number of transitions in the Petri Net.*

Proof.

Necessary. Assume a Petri Net is not safe ⇒ there exists at least one place that has more than one token at the same time, and since tokens represent signals in the system ⇒ there exists two instances of the same signal at the same time in the system having different event-based references ⇒ the system is not event-synchronous.

Sufficient. Given that the network is live, then deadlock would not occur. In addition, if a Petri Net is safe implies that it is 1-bounded, which means that the number of tokens in each place does not exceed 1 for any marking reachable from the initial marking, M_0 [83]. For the communication this implies that there can only be one instant of a signal at any point in the system. However this does not guarantee that other instances do not exist at other points. This is accomplished by requiring the system to be fair with a reproduction vector x_{nx1}, where $x = [1, 1, ...1]$ and n is the number of transitions in the Petri Net.

A reproduction vector, x, is a minimal nonzero T-invariant, where T-invariant is an n-vector $x \geq 0$ such that $A^T x = 0$. A is the incidence matrix of a Petri Net, where $A = [a_{ij}]$ is an nxm matrix of integers and its typical entry a_{ij} is given by

$$a_{ij} = a_{ij}^{+} - a_{ij}^{-}$$

where a_{ij}^{+} is the number of arcs from transition i to its input place j and a_{ij}^{-} is the number of arcs to transition i from its input place j [91]. An n-vector is a T-invariant if and only if there exists a marking M_0 and a firing sequence σ from M_0 back to M_0

with its firing count vector equal to x [91]. This simply means that a reproduction vector corresponds to the minimum number of times each transition has to fire for the initial marking to recur.

Since a fair network has at most one reproduction vector it implies that x_{nx1} is unique. Given that $x = [1, 1, ...1]$, then each transition has to happen once for the initial marking to recur. This also means that a transition cannot fire more than once before all the other transitions have fired because they are in a fair relation [91]. So a new instant of a signal will not be generated unless the earlier one has been processed. Therefore, this ensures the second requirement for event-synchronization.

◇

The Petri Net design procedure that results in event-synchronous systems is developed in the next chapter.

4.3 Comparison with Other Models and Approaches

There are some common methods used in the literature to model and analyze teleoperation systems, the most popular ones are networks, wave variables and dynamic equations. The modeling and analysis approach examined here is based on network modeling, which was used by Anderson in [4] [32].

Anderson bases his analysis on the fact that for the system to be stable it should be passive. Using the analogy between mechanical and electrical systems he represented a teleoperator as a network, as shown in Figure 4.11. In this model, the master, communication block, and slave are represented by two-port, as for the operator and environment they are represented by one-port. An n-port is characterized by the relationship between effort F (force, voltage) and flow v (velocity, current). This

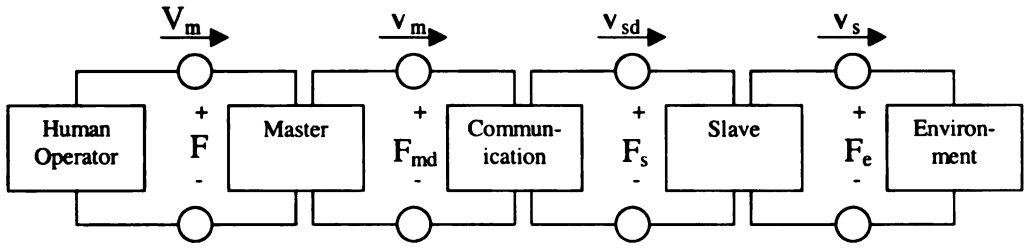

Figure 4.11: Network representation of teleoperator. The different terms are as explained in Table 4.1.

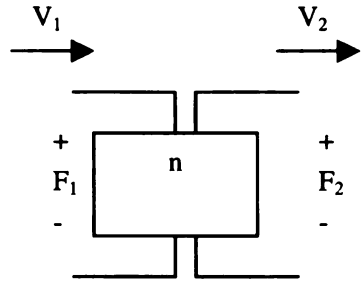

Figure 4.12: Transformer.

relationship for one-port is specified by its impedance $Z(s)$ according to

$$F(s) = Z(s)v(s) \tag{4.27}$$

where $F(s)$, $v(s)$ are the Laplace transforms of $F(t)$ and $v(t)$. As for the two-port, the relationship is

$$\begin{bmatrix} F_1(s) \\ -v_2(s) \end{bmatrix} = \begin{bmatrix} h_{11}(s)h_{21}(s) \\ h_{12}(s)h_{22}(s) \end{bmatrix} \begin{bmatrix} v_1(s) \\ F_2(s) \end{bmatrix} = H(s) \begin{bmatrix} v_1(s) \\ F_2(s) \end{bmatrix} \tag{4.28}$$

where $H(s)$ is the *hybrid matrix* and F_1, F_2, v_1 and v_2 are as defined in Figure 4.12.

For the system to be stable, its different components have to be passive. That is, they can dissipate energy but cannot increase the total energy of the system they are part of. This is why Anderson modeled all the components of the system using passive

circuit elements. Then to achieve a stable system under time delay, he choose a control law that would make the characteristics of the communication block identical to a two-port lossless transmission line, which is a passive element. The derived control law for the communication circuit is

$$F_{md}(t) = F_s(t-T) - v_{sd}(t-T) + v_m(t) \tag{4.29}$$

$$v_{sd} = v_m(t-T) - F_s(t) + F_{md}(t-T) \tag{4.30}$$

where T is delay, and the other terms are explained in Table 4.1. Comparing this approach and model to the ones developed in this research, several differences are found. The stability analysis in this case relies on the specific model used, whereas the stability analysis done for event-based control systems is independent of the specific system developed. Moreover, the model assumes constant delay, which is also the same in both directions of the communication, whereas no assumptions regarding the delay are made in the analysis and modeling developed in this document. In addition, the human model is assumed to be passive, which is not the case for the research conducted. Similar limitations and differences apply to the other modeling and analysis techniques proposed in the literature.

The next chapter presents the design procedures required to develop stable, event-transparent and event-synchronous real-time teleoperation systems.

CHAPTER 5

DESIGN OF EVENT-BASED REAL-TIME TELEOPERATION SYSTEMS

For real-time teleoperation systems to become commercially attractive they have to be safe and efficient. In other words, some performance measures, such as stability, event-transparency and event-synchronization have to be guaranteed. This chapter gives the design procedures required to accomplish this guarantee.

5.1 Design of Stable and Event-Transparent Teleoperation Systems

To achieve stability, the system has to be designed as an event-based control and planning system. That is the signals in the system have to be sampled with respect to an action reference other than time. Equivalently, this implies that the system components reference a non-time based action reference, referred to as event-based action reference.

For these systems to be stable, the event-based reference has to be a non-decreasing function of time as discussed in Section 3.2. As for event-transparency, additional conditions have to be satisfied. Note that, being an event-based control system gives

$$F_h(s) = \alpha F_e(s) \quad and \quad V_h(s) = \beta V_e(s) \tag{5.1}$$

$$\Rightarrow \frac{F_h(s)}{V_h(s)} = C\frac{F_e(s)}{V_e(s)}$$

where F_h and V_h are the supermedia and commands generated by the joystick re-

spectively. Similarly, F_e and V_e are the slave supermedia and control respectively and α, β and C are simply scaling factors, and s is the action reference. However, this condition is not enough for the system to be event-transparent. That requires the event-based reference, s, to be chosen in a way to ensure that $F_e(s)_{rd} = F_e(s)_{nd}$, where $F_e(s)_{rd}$ and $F_e(s)_{nd}$ are the supermedia sensed by the system under random delay and no delay conditions respectively. Additionally, the event-based sampling period, S, has to satisfy the following condition discussed in Section 3.3:

$$\frac{\pi}{KS} \geq w_1 \tag{5.2}$$

where K is the scaling between event-based reference and time, and where the supermedia detected is band limited by w_1.

The major difficulties in selecting s are the many uncertainties in the system. The main uncertainties are the path (trajectory) and the environment. The specific choice of s will depend on the type of feedback being used, the work space size (macro or micro) and the performance requirements. Typically, the event s is taken to be a function of the physical output; for example, the distance to the origin, the angle or the absolute position.

However, the event-based reference can also be a virtual one that does not correspond to any physical quantity. For example, it can be chosen to be the number of control cycles the system has performed. This can be achieved by using event-based planning and control, with the event taken as the command number. So if currently the operator is sending the n^{th} command, then the event is n. This choice of the reference is intuitive and does not require the operator to know what value it is, since the communication procedure will ensure the specific behavior of the system regardless of the operator's commands.

To describe event-based control and planning from signals perspective, the communication of control and feedback signals is described here. This discussion applies to all event-based systems regardless of the specific reference used. The sequence starts by the operator placing the joystick in a certain position that corresponds to a command. This position can be translated into a position or velocity command. The behavior of the joystick is similar to the gas pedal in a car, moving away from the center would generate a larger value for the command.

The command vector is then sent to the robot. This is the point where the choice of a specific reference would alter the behavior slightly. For the first case, where the reference is chosen to be the sequence number, once the command arrives it is immediately forwarded to the obstacle avoidance routine. If a command is not received for sometime the robot will time out and wait for a new command. As for the second case, where the reference is chosen to be a physical output, the command is not forwarded to the obstacle avoidance routine until the start of the next period. For example, if the reference was taken as the distance the robot traveled with a sampling period of two centimeters, then the new command will not be processed until the robot moves the two centimeters. If a command is not received before the robot had already moved the amount of the sampling period, the robot would be waiting for the new command and in this case the command is processed immediately.

Once the obstacle avoidance routine receives the command it will take decisions based on the results obtained from the proximity sensing units. This decision might be either to execute the command as is, to reduce the value and execute or not to move at all. The modified command is then sent to the local low level controller of the robot or robots.

Once this command is handed over to the controller the supermedia to be fed back is sensed and sent to the remote site. At the remote site each feedback type is given to the corresponding rendering device. For example, the joystick local controller

receives the force and the temperature rendering device receives the temperature. These devices render the supermedia and then the whole sequence is started again.

In this operation none of these step can be executed out of order, each event is triggered by the end of the previous one. What should be clear here is that during all this time, although the operator can move the joystick, the commands will not be sampled by the joystick until feedback is received and rendered. This ensures that every force the operator feels is the most recent one and that it corresponds to the current state of the robot. In addition, this procedure guarantees that the new command is the next one to be executed and no other commands are pending.

The next section gives a formal design procedure based on Petri Net that results in event-synchronous systems.

5.2 Design of Event-Synchronous Teleoperation Systems

Petri Net has proven to be a very beneficial tool for the modeling and analysis of different types of systems. Event-based control systems is one type where Petri Net can be used to model the communication and control logic. Both continuous and discrete aspects can be captured. More importantly, Petri Nets can be used to design and analyze different characteristics of the system. Event-synchronization is one of these performance measures that can be designed and tested using Petri Nets. To illustrate this, the following recursive design algorithm is proposed:

1. Make system live
2. Make system safe
3. Make system fair with reproduction vector x_{nx1}, where $x = [1, 1, ...1]$.
4. Check if system is still live, if not go to step 1.
5. Check if system is still safe, if not go to step 2.

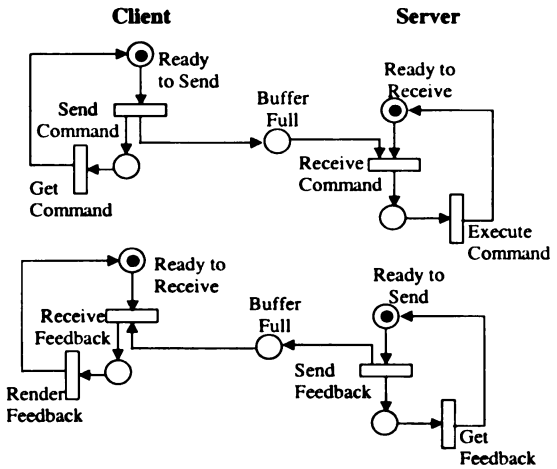

Figure 5.1: Petri Net model of a typical non event-synchronous teleoperation system.

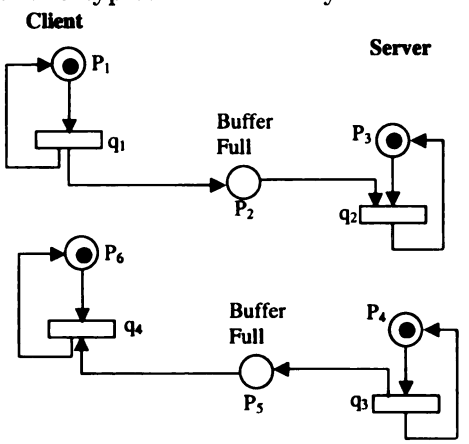

Figure 5.2: The equivalent reduced model of the system shown in Figure 5.1.

To demonstrate this design method the following representative examples are considered. The first example is that of a single operator single robot system and the second example is that of a multi-operator multi-robot system. Figure 5.1 shows the Petri Net model of a teleoperation system that is not event-synchronous. From the model it is clear that commands and feedback are sampled and transmitted independently of each other. This model can be reduced to simplify the design and analysis to the model shown in Figure 5.2.

It is clear that this system is not safe since transition q_3 can fire infinitely many times before transition q_4 fires even once and this will accumulate an infinite number

of tokens in place p_5. To satisfy safety and part of event-synchronization places p_2 and p_5 cannot have more than one token simultaneously. This condition can be written as

$$\mu_2 + \mu_5 \leq 1 \tag{5.3}$$

The model shown in Figure 5.2 can be modified to satisfy the condition in eq.[5.3] using Petri Net design procedures [92]-[94]. To modify the original net, referred to as the "process net", it is required to design a "controller net", which is made up of the process net's transitions and a separate set of places. Each condition to be satisfied will require one additional place in the controlled net called the "slack". The purpose of this slack is to receive the excess tokens from the places in the condition, thus ensuring that it is satisfied.

The result is a "controlled system" or "controlled net" with a composite change matrix D made up of both the process net composite change matrix D_p and the controller net composite change matrix D_c, where the composite change matrix is the transpose of the incidence matrix described earlier. So the composite change matrix of the process Petri Net shown in Figure 5.2 is

$$D_p = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.4}$$

with initial marking

$$\mu_{p0} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} \tag{5.5}$$

It is required to control this net so that the condition in eq.[5.3] is satisfied. This condition or constraint is of the general form

$$L.\mu_p \leq b \tag{5.6}$$

where μ_p is the marking vector of the Petri net, L is an n_cxn matrix, b is an n_cx1 vector and n_c is the number of constraints of the type shown in eq.[5.3] [92].

The "slack" or controlled places mentioned earlier transform the inequality constraints into equality conditions of the form

$$L.\mu_p + \mu_c = b \tag{5.7}$$

where μ_c is an n_cx1 vector which represents the marking of the controller places. Therefore, the specific case shown in eq.[5.3] can be written as

$$\mu_2 + \mu_5 + \mu_s = 1 \tag{5.8}$$

where in this case $L = [0\ 1\ 0\ 0\ 1\ 0]$, $b = 1$ *and* $\mu_c = \mu_s$.

The constraint described by eq.[5.8] represents a place invariant and must satisfy the place invariant equation

$$X^T \cdot D = [L \quad I] \cdot \begin{bmatrix} D_p \\ D_c \end{bmatrix} = 0 \Leftrightarrow \tag{5.9}$$

$$L \cdot D_p + D_c = 0 \Leftrightarrow \tag{5.10}$$

$$D_c = -L \cdot D_p \tag{5.11}$$

where I is an $n_c x n_c$ identity matrix [92]. In this case the controller net composite change matrix is $D_c = [-1\ 1\ -1\ 1]$. In addition, the initial marking of the controller net can be calculated using eq.[5.7] and the initial marking vectors:

$$L.\mu_{p0} + \mu_{c0} = b \Leftrightarrow \mu_{c0} = b - L.\mu_{p0} \tag{5.12}$$

which implies in this case that $\mu_{s0} = 1$. This results in a controlled network as shown in Figure 5.3 having the following composite change matrix and initial marking:

$$D = \begin{bmatrix} D_p \\ D_c \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 \\ -1 & 1 & -1 & 1 \end{bmatrix} \tag{5.13}$$

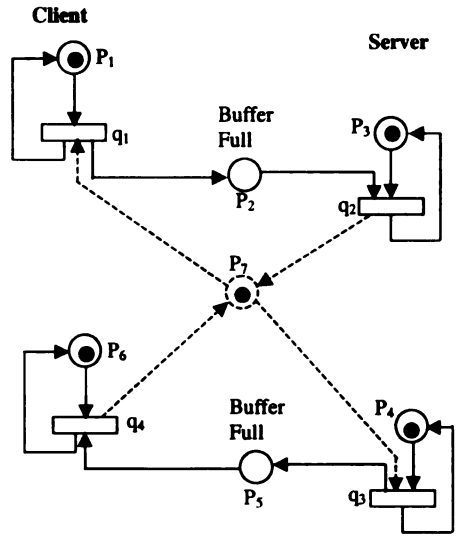

Figure 5.3: Petri Net model of the controlled net.

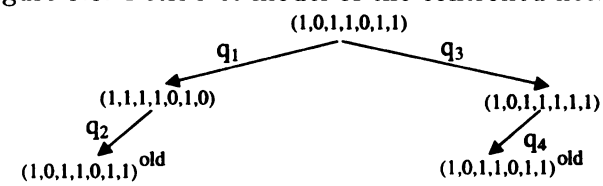

Figure 5.4: The coverability tree of the net shown in Figure 5.3.

$$\mu_{p0} = \begin{bmatrix} \mu_{p0} \\ \mu_{s0} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \tag{5.14}$$

The dotted part of the net represents the "controller" net, which is required for the "controlled" net to satisfy the constraint of eq.[5.3].

Since this model is bounded then the coverability tree can be used to study its liveness and safety [83]. The coverability tree of the net shown is Figure 5.3 is given

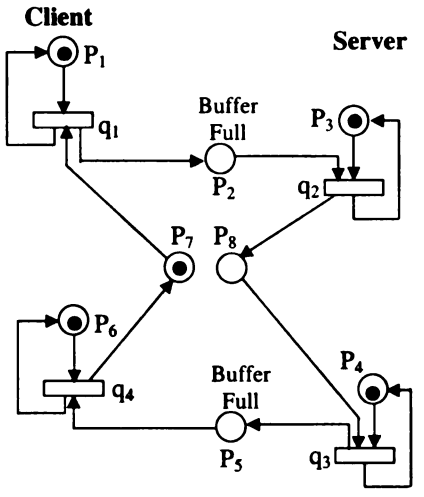

Figure 5.5: The modified net required to satisfy fairness.

in Figure 5.4. From this coverability tree it is clear that the system is live since there are no deadlocked markings and that the system is safe since all the markings have only zeros and ones in them [83].

To check if the system is fair with reproduction vector x_{nx1}, where $x = [1, 1, ...1]$, the necessary condition used is that for a net to be fair then if x is an n-vector of non-negative integers such that $A^T x \geq 0$ and $x \neq 0$, then every entry of x must be positive, where A is the incidence matrix. To test this, the incidence matrix of the system shown in Figure 5.3 is generated:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix} \tag{5.15}$$

It is easy to see that $x = [1\ 1\ 0\ 0]^T$ and $x = [0\ 0\ 1\ 1]^T$ result in $A^T x = 0$, so $x = [1\ 1\ 1\ 1]^T$ is not a reproduction vector. Also, not every entry of these vectors is positive so the net is not even fair. If the net was fair with a reproduction vector the

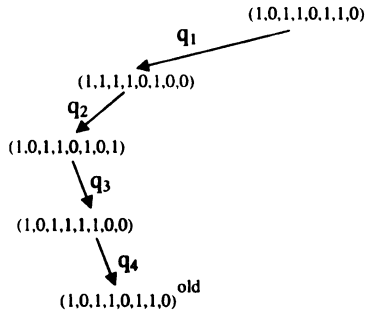

Figure 5.6: The coverability tree of the net shown in Figure 5.5.

incidence matrix A has to have a rank of $n-1$, where n is the number of transitions. This is not the case for this incidence matrix, which has a rank of 2.

In order to satisfy all these conditions and render the net fair it is modified to the net shown in Figure 5.5. The change proposed is to split place 7 into two places and thus increase the dimension of A.

To study the fairness of the system shown in Figure 5.5 its incidence matrix is constructed:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \end{bmatrix} \tag{5.16}$$

From this incidence matrix it is possible to solve for the reproduction vector $x = [x_1\ x_2\ x_3\ x_4]^T$ using $A^T x = 0$. This results in the following system of equations:

$$\begin{aligned} x_1 - x_2 = x_3 - x_4 = -x_1 + x_4 = x_2 - x_3 = 0 \\ \Rightarrow x_1 = x_2 = x_3 = x_4 \end{aligned} \tag{5.17}$$

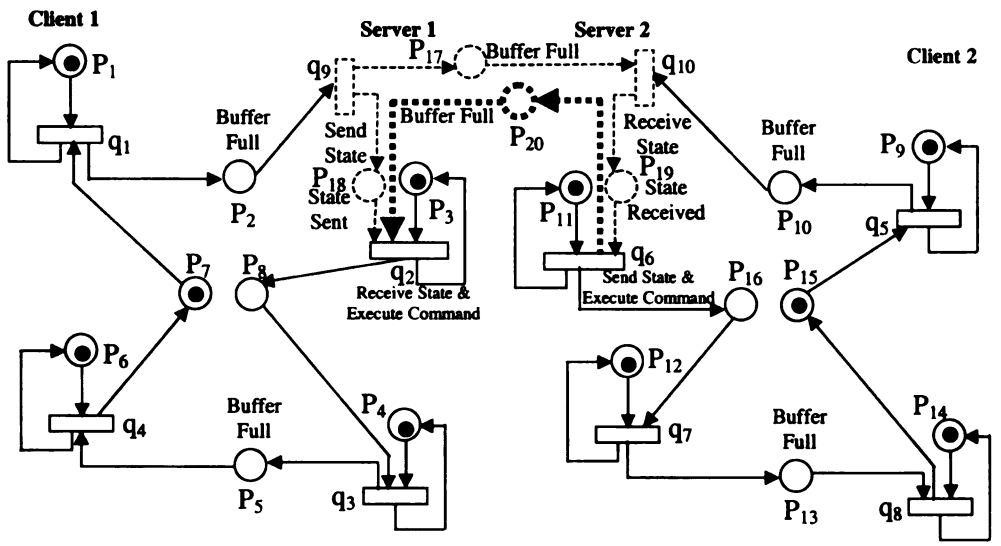

Figure 5.7: The Petri Net model at different design stages for an event-synchronous **multi**-operator multi-robot teleoperation system.

To check for fairness x is considered as an n-vector of non-negative integers with $x \neq 0$, which implies that $x_1 = x_2 = x_3 = x_4 = k$, where k is a positive integer. So **every** entry of x must be positive and since the network is bounded for any initial **marking** then the net is fair [91]. The reproduction vector is the minimal vector **satisfying** $A^T x = 0$ so for this system $k = 1$ and $x = [1\ 1\ 1\ 1]^T$. To further check, the **rank** of A is $n - 1 = 3$ and the reproduction vector is unique which are two properties **of fair** nets [91].

Checking liveness and safety properties is done using the coverability tree, which **is** shown in Figure 5.6. Since all the markings contain only ones and zeros then the **net** is safe, and since no marking is labeled "dead" then the net is live.

These steps illustrated using a Petri Net design procedure to obtain an event-**synchronous** single robot single operator teleoperation system. This system can be **further** developed to include multi-operators and multi-robots.

The model of such a system at different design stages is shown in Figure 5.7. **The** solid line parts model the original stand alone system, the dotted parts, which **include** transitions 9 and 10 and places 17, 18 and 19, are added at the first design

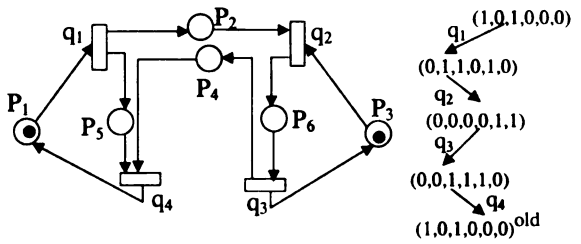

Figure 5.8: The resultant net from reducing the net shown in Figure 5.7 and its coverability tree.

stage and the bold dotted parts, which include place 20, are added at the second and final design stage.

Obviously the two stand alone systems are not event-synchronous since transitions in each fire irrespective of the other system's state and thus they are not fair. Therefore, there is a need for the two servers to communicate in order to convey state information. For this purpose transitions 9 and 10 and places 17, 18 and 19 are added to the net in order to send state information from one robot to the other. However, this is not sufficient because server 1 can still execute more times than server 2 in which case place 17 would have more than one token. This implies that the system is still not fair and even not safe. For this to be remedied server 2 is also required to send its state information, which is done by adding place 20. These additions compose a two-way handshake between the robots, which guarantees liveness, safety and fairness and thus event-synchronization.

To illustrate this the controlled net in Figure 5.7 is analyzed. First it is reduced to the net shown in Figure 5.8 with the coverability tree shown in the same figure. Since the coverability tree has no "dead" states and since it has only ones ad zeros then the net is live and safe.

To study the fairness of the system shown in Figure 5.7 its incidence matrix is constructed:

$$A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 1 \\
0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0
\end{bmatrix} \tag{5.18}$$

From this incidence matrix it is possible to solve for the reproduction vector $\boldsymbol{x} = [x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7\ x_8\ x_9\ x_{10}]^T$ using $A^T x = 0$. This results in the following system of equations:

$$
\begin{aligned}
x_1 - x_9 = x_3 - x_4 = -x_1 + x_4 = x_2 - x_3 = 0 \\
x_5 - x_{10} = x_7 - x_8 = -x_5 + x_8 = x_6 - x_7 = 0 \\
x_9 - x_{10} = -x_2 + x_9 = -x_6 + x_{10} = -x_2 + x_6 = 0 \\
\Rightarrow x_1 = x_2 = x_3 = x_4 = x_5 = x_6 \\
= x_7 = x_8 = x_9 = x_{10}
\end{aligned}
\tag{5.19}
$$

To check for fairness x is considered as an n-vector of non-negative integers with $x \neq 0$, which implies that $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = x_7 = x_8 = x_9 = x_{10} = k$, where k is a positive integer. So every entry of x must be positive and since the network is bounded for any initial marking then the net is fair [91]. The reproduction vector is the minimal vector satisfying $A^T x = 0$; so for this system, $k = 1$ and $x = [1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1]^T$. Therefore, the controlled system is safe, live and fair, which implies it is event-synchronous.

The design procedures discussed in this chapter constitute a general methodology for implementing real-time event-based teleoperation systems. This framework applies to any real-time teleoperation system regardless of the robot, the environment model, the human operator and the specific network used. Importantly, based on this design the system stability, event-transparency and event-synchronization are guaranteed regardless of the delay encountered.

The importance of these properties is demonstrated in the next chapter, which derives the conditions required to achieve safe and reliable multi-robot tele-coordination. The concepts of coordination for multiple robots and coordination index, which is a quantitative measure of coordination quality, are introduced. In addition, it is proved that event-transparency and event-synchronization are the requirements to achieve any level of tele-coordination feasible under no delay conditions.

CHAPTER 6

MULTI-ROBOT TELE-COORDINATION

Cooperating robots have been the topic of several research efforts ranging from autonomous manipulation to remote control [95]-[105]. In many scenarios, cooperation offers additional safety, efficiency and feasibility. For instance, an object might be too heavy or too fragile to be lifted by one robot only. Combined with the Internet, tele-cooperation provides the means for multi-experts at multi-sites to cooperate using multi-robots.

However, not all multi-robot cooperation systems, such as the one shown in Figure 6.1, are considered as coordinating systems. Coordination refers more specifically to the case where two robots are in contact with the same object or with each other. This mechanical contact results in close coupling between the robots and thus requires higher precision, stricter synchronization and a higher level of coordination.

Coordinating robots and more specifically tele-coordinating robots, where the robots are not autonomous but teleoperated, offer numerous technical difficulties. These difficulties relate to the synchronization of operation, coordination strategies and constraints on the manipulated object.

When it comes to tele-coordination, especially Internet-based, there are several additional challenges facing the achievement of synchronous operation, efficient coordination and safe manipulation. These challenges result from the randomness of time delay encountered over the Internet and the uncertainties in the environment, the object manipulated and the specific task path/trajectory.

Traditionally multi-manipulator coordination can be divided into centralized control [95] [96], where a single controller is supposed to control all the manipulators, and decentralized control [97] [98], where each robot is controlled by its own controller. The decentralized approach overcomes the computational burden imposed on the central controller. Multi-robot tele-coordination can be categorized similarly with

Figure 6.1: The general structure of a multi-robot tele-coordination system.

some approaches being a combination of both.

Tele-cooperation research found in the literature shares some or all of the following limitations; non-deterministic time delay is not considered, the control is not bilateral, the robots are not coordinating, there is no objective measure of the coordination, physically separated operators are not considered [99]-[102]. To overcome the challenges faced and the limitations found in the literature event-based planning and control is proposed for multi-robot coordination systems.

6.1 Modeling and Analysis of Tele-coordination Systems

First a measure of the tele-coordination level is established. In autonomous coordination, force has been the main measure of coordination. This has not been true for the case of tele-coordination. Tele-coordinating with multiple robots under certain force conditions is more difficult because of the uncertainties in the task and environment. Considering multiple robots that are teleoperated to coordinate in moving an object, the smaller the internal force that is felt by the manipulated object the better the coordination is. This translates into having the least opposition possible between the robots. To relate the internal force and the opposition between robots, consider the case of an object manipulated by two robots as seen in Figure 6.2 [103] [104]. The grasp is assumed to be rigid and the applied forces, f_1 and $f_2 \in \Re^3$, are represented

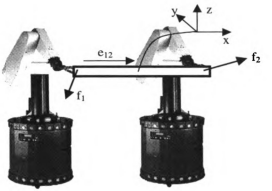

Figure 6.2: Two mobile manipulators moving an object.

in a coordinate system fixed in the object frame and whose x-axis is along the object. Let e_{12} be the unit vector along the link from grasp 1 to grasp 2. Let f be the 6x1 force vector

$$f = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \tag{6.1}$$

When the object is in static equilibrium the internal force, t, which is along the link between the grasps, and the applied forces, f, are related by

$$f = Et$$

where $E = [-e_{12}\ e_{12}] = [-1\ 0\ 0\ 1\ 0\ 0]^T$. When the object is not in static equilibrium the applied forces can be decomposed into two parts. A part which results in motion, f_e, and a part which results in tensions or internal forces, f_{in}. Therefore,

$$f = f_{in} + f_e \tag{6.2}$$

where $f_{in} = Et$. Solving for t results in

$$t = \overline{E}(f - f_e) \tag{6.3}$$

where $\overline{E}$ is a left inverse [103]:

$$\overline{E} = (E^T E)^{-1} E^T \tag{6.4}$$

For the two robot case $\overline{E} = \frac{1}{2}[-1\ 0\ 0\ 1\ 0\ 0]$. By definition f_e does not result in any tensions so $\overline{E} f_e = 0$ and the internal forces are given by

$$t = \overline{E} f \tag{6.5}$$

The general case of multiple robots is analyzed similarly and is discussed in [103]. From eq.[6.5] it is clear that

$$t = \frac{1}{2}(-x_1 + x_2) \tag{6.6}$$

where x_1 and x_2 are the x components of f_1 and f_2 respectively. From eq.[6.6] it can be inferred that

$$||t|| \leq \frac{1}{2}(max(x_1) + max(x_2)) \tag{6.7}$$

For coordination it is required to have a certain limit on the object's internal force, which can be achieved by limiting x_1 and x_2 as seen in eq.[6.7]. Since $max(x_1) = max(||f_1||)$ and $max(x_2) = max(||f_2||)$ then this limit can be accomplished by limiting the applied forces. In addition, since the magnitude of a vector in any frame is the

same then the applied forces can be limited in each robot's hand frame. However, in order not to limit the acceleration, only f_{in} need to be limited and not the total f.

Therefore, the maximum external force each robot senses f_{in}, not including force due to object weight and acceleration, can be considered as an objective measure of the coordination quality or level. The smaller the external force sensed the better the coordination, which in the ideal case translates into zero forces being detected. However, this ideal case even in autonomous operation is not feasible because of disturbances and errors. An additional intuitive condition is that the task should be accomplished, because there is no value in considering coordination quality if the task is not accomplished. This concept can be formally stated as,

Definition 4 *Coordination:*

n robots are coordinating with tolerance ϵ *if* $max(|f_{ei}(s)|) \leq \epsilon$ *and the task is accomplished, where* s *is the action reference,* $\epsilon \in \{\Re^+, 0\}$ *is called the coordination index and* $f_{ei}(s) \in \Re^3$ *is the external force sensed by robot* i, $(i = 1, 2, ..., n)$. $f_{ei}(s)$ *is given by* $f_{ei}(s) = F_i(s) - m_i a_i(s)$ *for* x *and* y *components and by* $f_{ei}(s) = F_i(s) - m_i a_i(s) - m_i g$ *for* z *component.* $F_i(s) \in \Re^3$ *is the total force detected by robot* i, m_i *is the mass carried by robot* i, $a_i(s) \in \Re^3$ *is the acceleration of robot* i *and* g *is the gravitational constant.*

If $\epsilon = 0$ *the robots are said to be executing ideal coordination.*

Therefore, to establish a certain level of coordination while guaranteeing task completion is a matter of limiting the force each robot is subjected to. This is similar to limiting the force each robot can execute. The force limited is the external force exerted on the robot excluding the weight of the object manipulated and the inertial forces.

Note that it is assumed the object is held by the robots with rigid grasp. This implies that there is no relative motion between the robots' grippers and the object.

Also the weight carried by each robot is assumed to be constant and no weight redistribution occurs. This is not a limiting condition for objects with uniformly distributed mass. As for objects that have non-uniformly distributed mass, a real-time re-calibration algorithm should be devised based on initial weight carried, initial posture, and changes in the posture.

To accomplish any specified coordination index feasible under no delay conditions the robots have to satisfy certain specifications as described in the next theorem.

Theorem 5 *If n robots are event-transparent and event-synchronous then they can be teleoperated under random delay conditions to coordinate to any tolerance ϵ, which is feasible under no delay conditions, where ϵ is the required coordination index.*

Proof. Let $V(s)$ be the path of the manipulated object required to accomplish the task. Corresponding to this path, under no delay conditions, each robot will have a path referred to as $V_{ndi}(s)$. $V_{ndi}(s)$ is the path of robot i required to accomplish the task under no delay tele-coordination conditions, with coordination index ϵ_{nd},

$$\Rightarrow max(|f_{ndei}(s)|) \leq \epsilon_{nd}$$

where $f_{ndei}(s)$ is the external force sensed by robot i under no delay condition and given by

$$f_{ndei}(s) = \begin{bmatrix} F_{ndxi}(s) - m_i a_{ndxi}(s) \\ F_{ndyi}(s) - m_i a_{ndyi}(s) \\ F_{ndzi}(s) - m_i a_{ndzi}(s) - m_i g \end{bmatrix} \tag{6.8}$$

where $F_{ndi}(s)$ is the total force detected by robot i under no delay conditions, m_i is the mass carried by robot i, $a_{ndi}(s)$ is the acceleration of robot i under no delay conditions and g is the gravitational constant.

Since the robots are event-transparent [106],

$$\Rightarrow f_{rdei}(s) = f_{ndei}(s)$$

where $f_{rdei}(s)$ is the external force sensed by robot i under random delay condition and given by

$$f_{rdei}(s) = \begin{bmatrix} F_{rdxi}(s) - m_i a_{rdxi}(s) \\ F_{rdyi}(s) - m_i a_{rdyi}(s) \\ F_{rdzi}(s) - m_i a_{rdzi}(s) - m_i g \end{bmatrix} \tag{6.9}$$

where $F_{rdi}(s)$ is the total force detected by robot i under random delay conditions, m_i is the mass carried by robot i, $a_{rdi}(s)$ is the acceleration of robot i under random delay conditions and g is the gravitational constant.

$$\Rightarrow max(|f_{rdei}(s)|) = max(|f_{ndei}(s)|) \leq \epsilon_{nd} \tag{6.10}$$

Also from even-transparency it implies that $V_{rdi}(s) = V_{ndi}(s)$ [106], where $V_{rdi}(s)$ is the path of robot i under nondeterministic delay tele-coordination conditions. Since the system is event-synchronous it implies that all the robots are using the same reference s and thus all $V_{rdi}(s)$ are executed synchronously with respect to this reference. Therefore, the i robot paths $V_{rdi}(s)$ (i=1, ...,n) will result in the same object path, $V(s)$. So the task is accomplished with a coordination index deduced from eq.[6.10] and equal to ϵ_{nd}.

$\Diamond$

Corollary 6.1.1 *n event-transparent and event-synchronous robots can ideally coordinate* ($\epsilon = 0$).

This is possible as long as ideal coordination is feasible, which might not be realistic in most scenarios. Even if humans are coordinating to move an object ideal coordination is very difficult to achieve because this will result in indefinite waiting. One of the humans has to move initially and since perfect synchronization is not possible this will cause some force on the other humans hand.

The control strategy required to achieve coordination with a certain tolerance level is the topic of the next section.

6.2 Control of Tele-coordination Systems

A control strategy is required to attain coordination with a certain tolerance. This is accomplished for small tolerance by the following set of control laws:

- For all i, stop moving if $max(|f_{ei}(s)|) \geq \epsilon$.
- In case $max(|f_{ei}(s)|) \geq \epsilon$ for any i, then for all i, if $V_i(s)$ is in the direction of reducing f_{ei} execute it else discard it.
- Execute $V_i(s)$ only when all $V_i(s)$ are received for all i.

The first law will guarantee that the robot will not cause its force to exceed the required coordination index. This is sufficient in case two robots only are coordinating because in this case what one robot is detecting is just the same as the other but in the opposite direction. However, in the case of more than two robots, other robots have to take into consideration the forces sensed by all the robots coordinating because in this case it is possible for only one robot to reach the coordination index but not for the others. This is accomplished using the second law, which requires all the robots to operate only in the decreasing force direction even if one robot has reached the limit. For this to be accomplished each robot has to communicate its force sensed to the other robots or at least inform the other robots that it has reached its force

limit. This way the other robots would not execute new commands unless they are in the decreasing direction of force, which can be simply calculated using a dot product between the force sensed and the command desired. Explicitly communicating force information is not required in the two robots case, where it is sufficient for each robot to base its movement decisions on its own state.

These two laws will guarantee that

$$max(|f_{ndei}(s) - f_{rdei}(s)|) \leq 2\epsilon$$

Since ϵ is considered to be relatively small,

$$\Rightarrow f_{rdei}(s) \rightarrow f_{ndei}(s)$$

Based on event-transparency $\Rightarrow V_{rdi}(s) \rightarrow V_{ndi}(s)$.

This implies that each robot executes similar paths with respect to the event-based reference under no delay and random delay conditions. However, in order for the object to follow its desired path too the robots have to be event-synchronized. This is accomplished by the third law, which requires all the robots to utilize the same reference.

This control strategy will result in the operator controlling the robot as long as the force detected is less than the coordination index. Once this limit is reached by one of the robots they would allow only movement that decreases this force.

The next chapter deals with the implementation and experimentation. Implementation details of several developed systems are given. Also the experimental results and their implications are discussed.

CHAPTER 7

IMPLEMENTATION AND EXPERIMENTATION

To illustrate the generality of the design methodology and the theory developed, several different teleoperation systems were implemented and tested. This chapter presents the implementation details of these systems along with their experimental results. Most of these systems share common hardware and software architecture and specifications, which are discussed in Section 7.1 and Section 7.2. Section 7.3 gives the specification and performance of the Internet connections utilized during the different experiments. The sections following these discuss the specific systems and their experimental results. Section 7.4 discusses the teleoperation of a mobile robot giving the implementation details and several experimental results. Section 7.5 and Section 7.6 present a mobile manipulator teleoperation system using velocity and position control respectively. Section 7.7 and section 7.8 cover the implementation and experimentation of multi-site multi-operator tele-cooperation and tele-coordination systems respectively. Section 7.9 gives the implementation and experimentation of a tele-autonomous robotic formation system. Section 7.10 details a supermedia enhanced teleoperation system with its experimental results. Section 7.11 illustrates the generality of the developed approach by describing the implementation and experimentation of a micro manipulator teleoperation system.

7.1 General Hardware Architecture and Specifications

The diversity of the hardware used forced a careful study of integration and interfacing. As seen in Figure 7.1, which shows the general hardware architecture, and Table 7.1, which gives the specifications of the hardware components, the system consists of different operating systems and different configurations. So the problem of interconnection had to be studied carefully.

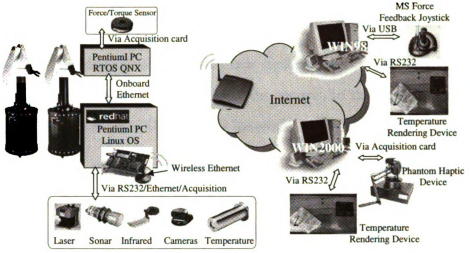

Figure 7.1: The hardware architecture shared by most of the systems developed.

Device	Specification
Joysticks	Microsoft SideWinder Force Feedback Pro. 3 degrees of freedom (dof) position, 2 dof force rendering
Haptic devices	Phantom 1.5 and Phantom Desktop 6 dof position, 3 dof force rendering
Local PCs	Pentiums WIN98, WINXP, WINNT, WIN2000 Interfaces to the joystick via USB port, to the haptic devices via acquisition cards, and to the temperature rendering device via the serial port.
Access Point	Proxim RangeLan2 radio access point Connects the robots to the Internet
Mobile Robots	Nomadic XR4000, 2 Pentium PCs on board Linux and QNX Interface to the Internet via wireless ethernet Equipped with multiple types of sensors: Infrared, Ultra sonic, Laser, Bumper and contact/non-contact temperature sensors.
Manipulators	PUMA 560 Equipped with force/torque sensors
Cameras	Sony EVI-D30 cameras and X10 wireless camera
Temperature Sensing and Rendering System	Designed and implemented in house

Table 7.1: Specifications of most of the hardware used.

The hardware used in most of the experiments included all or part of the following:

- Joysticks: Programmable force feedback joysticks, which are used to obtain commands and render force. The joysticks connect to the local PC, which is running WIN98 or WINXP operating systems, via the USB or the game port.

- Haptic Devices: Phantom devices, which are used to obtain position commands and render force. These devices interface to the local PC, which is running WINNT or WIN2000, via acquisition cards.

- Local PCs: Used to interface to the joysticks, haptic devices and temperature rendering device and used to control them. Also used to communicate over the Internet with the robots. In addition, they are used to display real-time video streams.

- Access point: Creates the link between the Internet and the robots via wireless communication.

- Mobile Robots: The XR4000 mobile robots used are equipped with several sensors (infrared, ultrasonic, laser and bumper), which are used to detect and avoid obstacles, and they are equipped with contact and non-contact temperature sensors. The robots execute commands depending on the surrounding environment and based on an obstacle avoidance algorithm discussed later in this chapter.

- Manipulators: The PUMA560 manipulators used are equipped with 6 dof force/torque sensors.

- Cameras: The robots have a camera mounted on-board to allow visual feedback to the operator via the Internet. Also other cameras are mounted overhead to provide a global view of the environment. These cameras can be controlled with the joystick to alter pan, tilt and zoom.

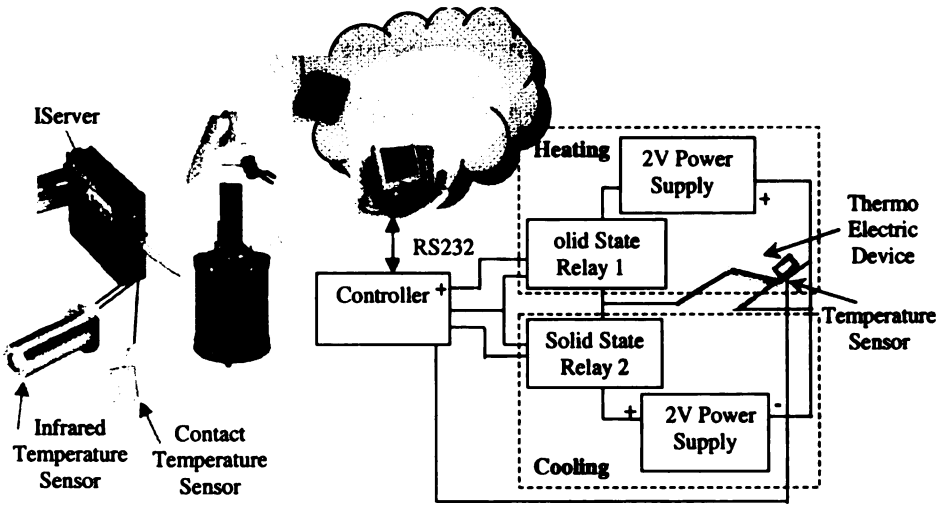

Figure 7.2: The remote temperature sensing and rendering system architecture.

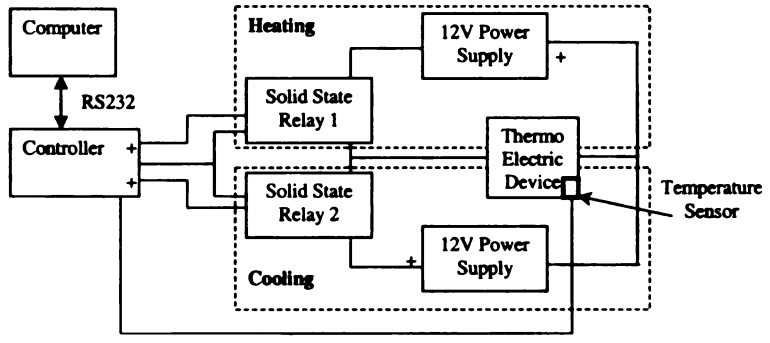

Figure 7.3: The architecture of the temperature rendering device developed.

- Temperature Sensing and Rendering System: The remote temperature sensing and rendering system architecture is shown in Figure 7.2. The temperature is sensed in real-time using contact or infrared non-contact temperature sensors. These sensors interface to the robot via the Iserver, which is an internet embedded device. The robot then relays the temperature information to the rendering device via the Internet.

A temperature rendering device was developed to integrate within the Internet-based teleoperation system. This device, which has the architecture shown in Figure 7.3, is based on thermoelectric technology. Thermoelectric coolers/heaters are solid state heat pumps that operate on the Peltier effect. The Peltier effect states that there

is a heating or cooling effect when electric current passes through two conductors. A voltage applied to the free ends of two dissimilar materials creates a temperature difference. Therefore, the temperature on one end can be controlled by controlling the voltage and its polarity across the device. The controller interfaces with the computer via the serial port and controls the voltage across the device. The polarity is controlled based on whether cooling or heating is required and is done by switching between two circuits. A contact sensor is mounted on the device in order to feedback the actual temperature and attain closed loop control.

7.2 General Software Architecture and Specifications

This section covers the general software architecture, which is shown in Figure 7.4, detailing the specifications of most of the routines developed, which were programmed using C++ and Microsoft Visual C++. In addition, the obstacle avoidance routine, which was used in most of the developed systems, is examined and the virtual force generation algorithm is presented.

The main difficulty in the software development was to have different operating systems (Linux, QNX, WIN98, WINNT, WIN2000, WINXP) communicate with each other. Here came the role of the communication protocol TCP/IP, which is supported by most operating systems. So the Internet was not only acting as a communication link but also as a translator.

In Internet communication usually there are two types of processes, servers and clients. Servers, as the name suggests, provide a service to clients. Typically, servers would run on the robot and clients would run on the remote machines. In the implementations done, TCP was used for communication between the clients and servers and UDP was used for video transmission. TCP had to be used to guarantee the arrival of packets since TCP handles retransmission of lost packets. If this was not the case, then lost packets will result in a deadlock, where either the client is waiting

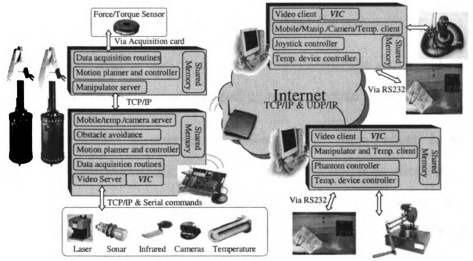

Figure 7.4: The software architecture shared by most of the systems developed.

for the server or vise versa because of the synchronization between them. This is handled by TCP at a lower level than the application and is transparent to the operator. Also, haptic and temperature information does not require a large bandwidth and while TCP is acknowledging the reception of a packet the processes can go on with the execution; therefore, the overall performance of the system is not affected. In case TCP is not available on a certain operating system this will have to be handled at the application level.

The following describes the specifications, operation and interaction of the servers, clients and general routines developed and shown in Figure 7.4. Note that this is a general architecture and was altered to fit the specific implementations. Therefore, some of these routines were not used in all systems or they might have been combined in one process or divided into several. These routines and their specifications are:

- Data Acquisition Routines: These routines run on the PCs controlling the mobile and the manipulator. They are used to acquire data from the force/torque sensors, laser, sonar, infrared and temperature sensors. This data is then used by other routines for feedback and closed loop control. These routines interface to the different sensors via acquisition cards, serial ports or ethernet cards.

The following routines run on the PC controlling the manipulator:

- PUMA Motion Planner and Controller: This controls the arm based on either velocity or position commands depending on the system used.

- Manipulator Server: This server receives commands from a client. In addition, it forwards commands to the PUMA motion planner and controller and forwards commands to the mobile server in case the mobile manipulator is used. The communication with the clients and mobile server are done using TCP/IP. This server also feeds back force information to the clients.

The following routines run on the PC controlling the mobile:

- Mobile/Temperature/Camera Server: This server receives commands either from a client or from the manipulator server using TCP/IP. Depending on the system, this server either controls the camera or forwards commands to the mobile motion planner and controller. It also feeds back force and temperature information.

- Mobile Motion Planner and Controller: This controls the mobile based on either velocity or position commands depending on the system considered.

- Obstacle Avoidance: This routine runs on the mobile and will be discussed in details in the following subsection.

- Video Server/Video Conferencing Tool (VIC): The video server tags each frame with the event reference to be used by the video client for event-synchronization. Then these frames are sent to the client via UDP using Video Conferencing Tool (VIC). This video conferencing application was developed by the Network Research Group at the Lawrence Berkeley National Laboratory in collaboration with the University of California, Berkeley. VIC was used to transmit video feedback from the on-board and the overhead camera to the operator.

The following routines run on the remote PCs connected to the joystick and to the temperature rendering device:

- Video Client/Video Conferencing Tool (VIC): This client is used to event-synchronize frames, which as received by VIC, with force and temperature.

- Mobile/Manipulator/Camera/Temperature Client: This client is used to interface with the MS joystick. It can be used to command mobile or mobile manipulator velocity or to command the camera. It communicates with the Mobile/Camera server using TCP/IP. It also receives the force and temperature from the servers and forwards them to the joystick controller and the temperature device controller respectively.

- Joystick Controller: This is responsible for controlling the joystick and rendering the force received.

- Temperature Device Controller: This is responsible for controlling the temperature on the rendering device via RS232.

- Manipulator and Temperature Client: This client is used to interface with the Phantom haptic device. It is used to command the manipulator position and receive force and temperature feedback using TCP/IP.

- Phantom Controller: This is responsible for controlling the phantom and rendering the force received.

The different routines running on the same machine communicate via shared memory. Note that several of these routines are in practice either combined or split into different processes. In addition, not all of them are used in all the systems developed.

Figure 7.5: The XR4000 mobile robot.

Obstacle Avoidance and Virtual Force Generation

Obstacle avoidance is an important safety feature in teleoperation systems. Because of delay, teleoperated robots need to have a certain level of intelligence. Since what the operator sees is not the most recent robot state, collisions might occur. These collisions can be avoided by employing an obstacle avoidance algorithm on the robot. A detailed explanation of the sensors available and their distribution on the robot is given. Then their use for obstacle avoidance and virtual force generation is discussed.

As seen is Figure 7.5, the XR4000 is a barrel shaped mobile robot. It is equipped with four types of sensors:

- Tactile Sensors (Bumper): They provide information about the physical contact with the environment. The Nomad XR4000 has 48 bi-level tactile sensors that surround its top and bottom perimeters. Additionally, it has 4 door bumpers on each door that sense contact between the top and the bottom perimeters. The sensors are divided into 6 sets, 2 per each of the 3 doors. Figure 7.6 gives the distribution and numbering of these sensors [107].

- Infrared Proximity Sensors: Give a range information to nearby objects (typically less than 30 to 50 cm away). The range is determined by emitting infrared

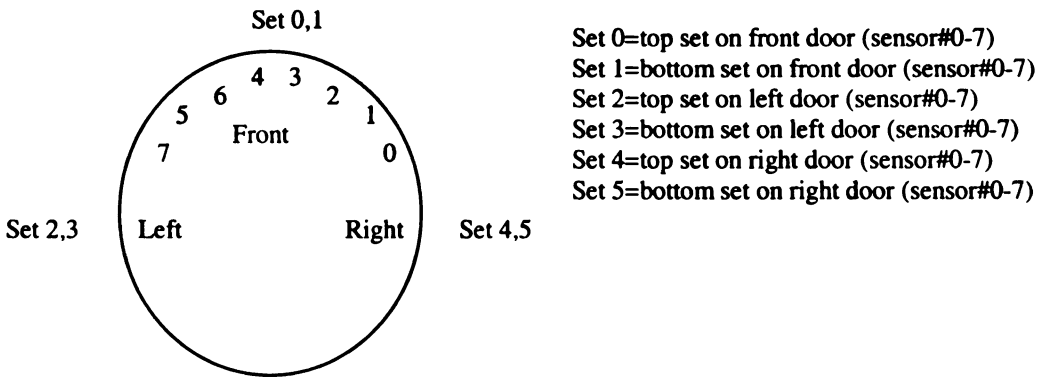

Figure 7.6: A top view of the robot that gives the distribution and numbering of the different sensors.

energy using high-current LEDs and sensing the amount of returned energy with infrared photodiodes. Note that the returned energy is a function of the object's reflectivity, that is why these sensors perform poorly in distance measurements. The robot has 48 of these sensors distributed just like the tactile sensors in 6 sets, and they have the same numbering shown in Figure 7.6 [107].

- Sonar Proximity Sensors: These provide range information to objects that are relatively far away (between 15 and 700 cm away). The distance is calculated by multiplying the speed of sound by the time of flight of a short ultrasonic pulse traveling to and from a nearby object. The robot has 48 of these sensors distributed just like the others, in 6 sets, and they have the same numbering shown in Figure 7.6 [107].

- Laser Proximity Sensor: This is a Sensus 550 "time of flight" laser range finding system. It provides 180 degrees of distance information in a plane at 0.5 degree increments with a scanning rate of $20Hz$. This sensor is located inside the robot and fires from an opening in the front door.

Obstacle avoidance is accomplished according to the flow chart in Figure 7.7. First, all the bumper sensors are checked, which gives 360 degrees coverage around the robot. If any is hit, the motion is stopped and a flag is set and sent back to the

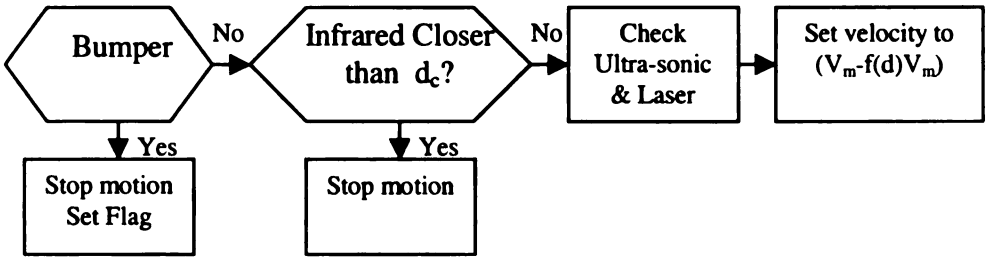

Figure 7.7: Flow chart for deciding velocity based on sensors.

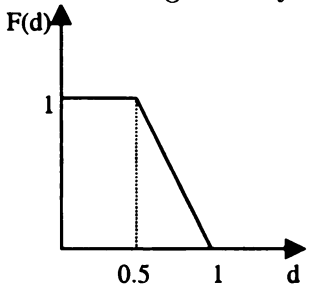

Figure 7.8: The fraction of velocity that is deducted based on the distance d in meters to the closest obstacle.

joystick, where a sudden jerking force is played to convey that a bump has occurred. If none is hit, the infrared sensors, 90 degrees from both sides of the desired motion, are checked. If any detects an object closer than d_c, which is a predefined programmable critical distance, the motion is stopped. If neither case is true, the ultra-sonic sensors are checked along with the laser. Again, 90 degrees from both sides of the desired velocity are checked. Based on the one that gives the closest distance, d, the velocity is set to V_s, which is given in eq.(4.6), where $V_{in} = f(d)V_m$. So the velocity set would be a fraction of the desired one based on the predefined programmable function $f(d)$. $f(d)$ can be any function of distance depending on the safety requirements. In most implementations $f(d)$ is as shown in Figure 7.8.

According to this specific function, the robot will slow down linearly when an object is detected between 0.5 meter and 1 meter, then it would stop if an object is detected closer than 0.5 meter. As seen in Figure 7.9, two regions will exist around the robot, one that would cause the robot to slow down and another to completely

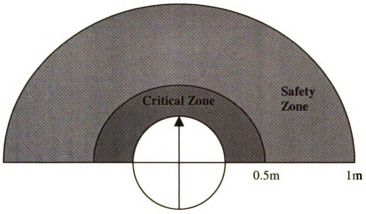

Figure 7.9: The two safety regions around the robot according to the direction of motion.

stop.

In case virtual force is desired, it is generated by calculating the difference between the actual velocity of the robot and the desired one. This difference is fed back to the joystick where it is rendered. So the higher the tracking error the higher the force is, which enables the operator to deduce the distance to obstacles in the direction of motion. This gives the benefit of sensing the obstacle without contact and before the visual feedback conveys it since the force feedback is faster than visual feedback. However, not all the systems developed included this virtual force since actual force was used.

7.3 Specification and Performance of Internet Connections

Concerning the networks that were used for experimentation is this project, there were three main communication setups: 1) a station in the Robotics and Automation laboratory at Michigan State University was used to control the robot, 2) a machine in the Robot Control laboratory, at the Department of Automation and Computer-Aided Engineering at the Chinese University of Hong Kong[1], was used, 3) a machine in the Center for Cooperative Research in Advanced Science and Technology at Nagoya

[1] distance between East Lansing, where the robot was, and Hong Kong is about 7824 miles

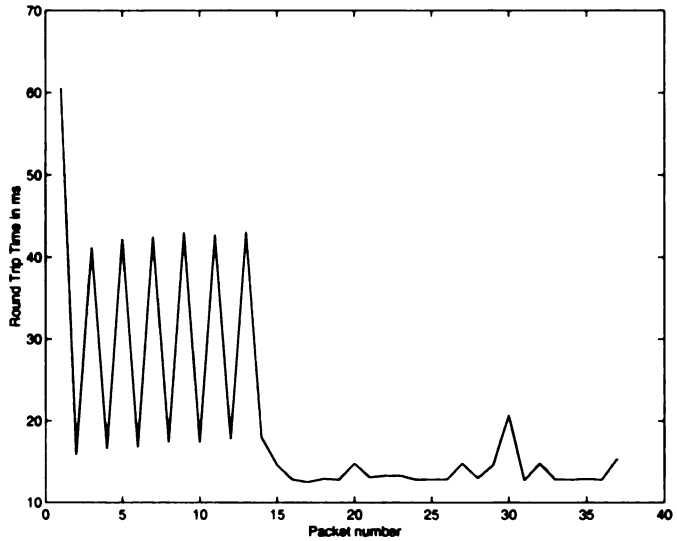

Figure 7.10: Round trip delay in milliseconds for packets transmitted between the robot and the MSU station.

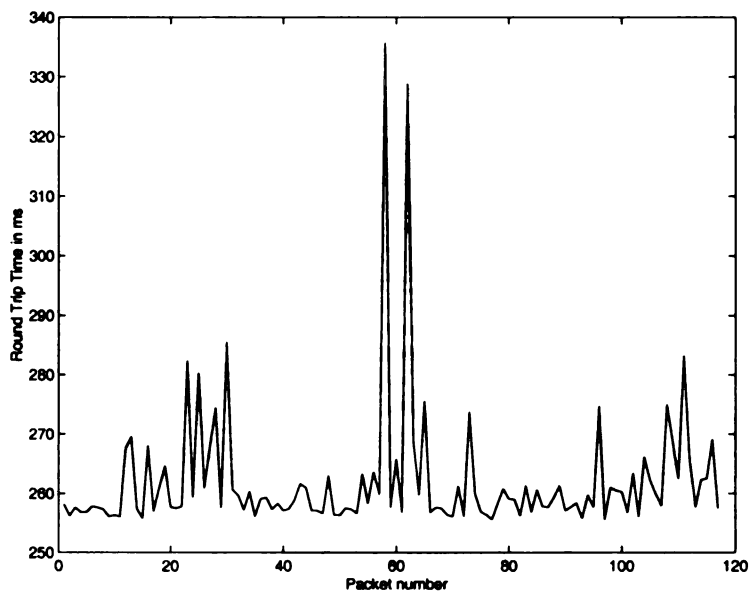

Figure 7.11: Round trip delay in milliseconds for packets transmitted between the robot and the Hong Kong station.

University[2], was used.

The robot uses wireless ethernet, which has a transmission rate of $1.6Mbps$. The remote machines either use 10Base-T ethernet, which has a transmission speed of

[2]distance between East Lansing, where the robot was, and Nagoya, Japan is about 6464 miles

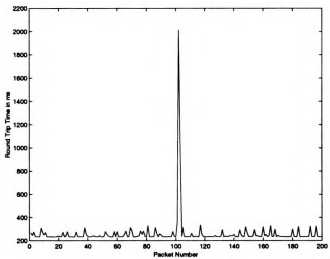

Figure 7.12: Round trip delay in milliseconds for packets transmitted between the robot and the Japan station.

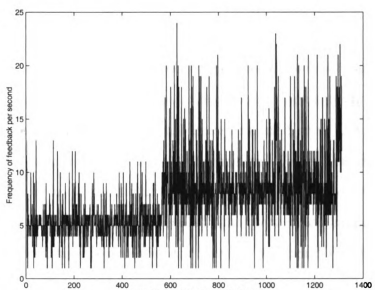

Figure 7.13: The frequency of control and feedback signals between the robot and the Hong Kong station.

$10Mbps$, or they use ISDN which ranges from $56Kbps$ to $128Kbps$.

The round trip delay between the robot and the teleoperating stations at MSU, Hong Kong and Japan is plotted in Figure 7.10, Figure 7.11 and Figure 7.12, respectively. As for Figure 7.13, it shows the frequency of operation; that is, the number of times per second that the robot receives a command and the operator feels force

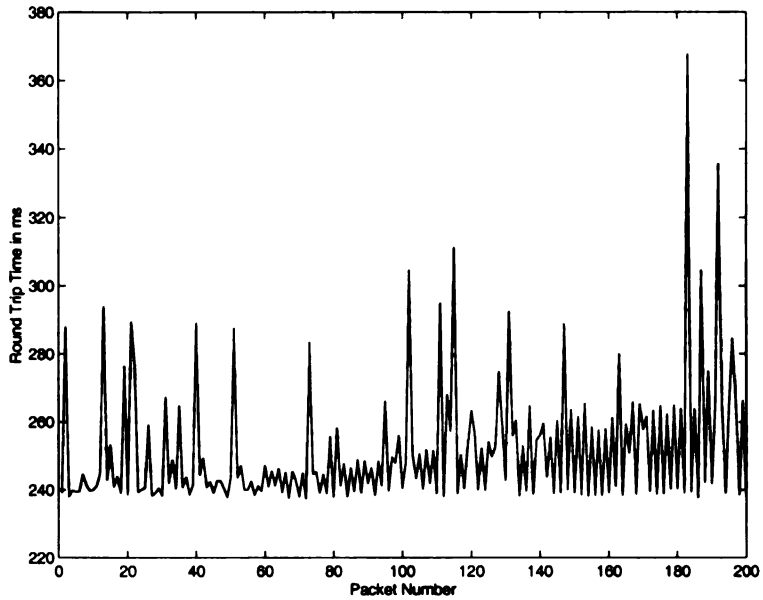

Figure 7.14: Round trip delay in milliseconds for packets transmitted between the robot and the Hong Kong station.

feedback during actual real-time control. From these plots it is clear that the delay is *random* and cannot be easily predicted; therefore, any assumption made about the time delay will be substantially limiting. Figure 7.11 and Figure 7.14 give plots of the round trip delay in milliseconds during different days. From these figures note that delays change according to several factors: time of day, what day and what time of the year the experiment is being done. Meaning that these delays cannot be predicted, which makes the real time control over the Internet significantly more difficult.

7.4 Mobile Robot

The general form of a mobile teleoperation system is seen in Figure 7.15. In this system the operator sends velocity commands and the remote machine feeds back force and video information. This section presents the implementation of an event-based teleoperation system for a mobile robot. In addition, the experimental results of several scenarios are discussed.

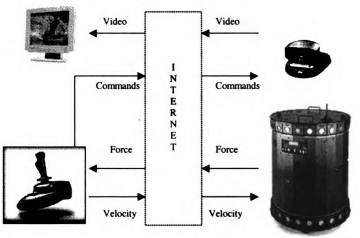

Figure 7.15: The general structure of the mobile teleoperation system developed.

7.4.1 System Implementation

Implementation is divide into two parts: hardware and software. The hardware section details the different components used, their specifications and inter-connections. As for the software section, the different modules and their interactions will be examined.

Hardware

The hardware architecture of this system is shown in Figure 7.16. The joystick used is the MS SideWinder Force Feedback Pro. and the mobile robot is a Nomadic XR4000. The components used are similar to the ones discussed in the general case.

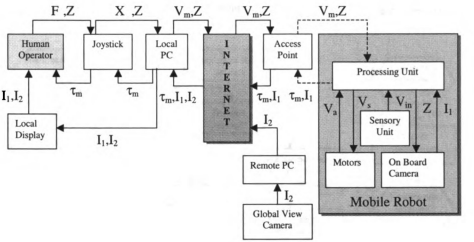

Figure 7.16: The hardware architecture of the mobile teleoperation system.

Software

The main software developed can be divided into three main parts: mobile server, camera server and mobile/camera client.

Mobile Server: In this system, the main service is moving the robot and sending feedback. As seen in Figure 7.17, which is a flow chart of the mobile server, the mobile/camera client and the camera server, this is mainly what the mobile server does. Except that the server does not execute the request blindly, it first checks the proximity sensors and based on that a decision is made according to the obstacle avoidance algorithm flow chart shown in Figure 7.7 and discussed previously.

After the velocity to be set is decided, it is sent to the local controller for execution. Then the server checks the actual velocity of the robot and subtracts that from the original desired one V_m and sends it back to the client as force feedback. In the case where the action reference is chosen to be the control sequence number, the server would execute the velocities for 250 milliseconds; after which, if no new command is received, it would time out and stop moving waiting for the next command. On the other hand, in the case where the action reference is chosen to be the distance traveled by the robot, the commands are not updated until the robot moves the

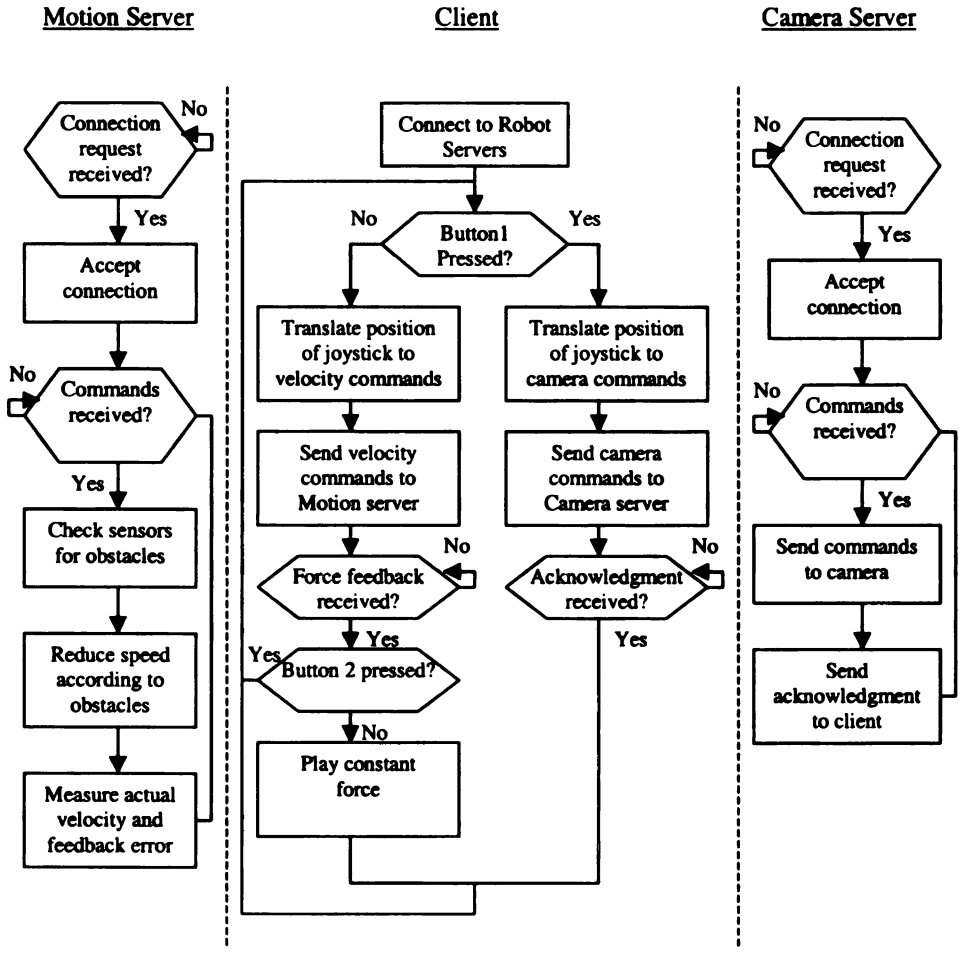

Figure 7.17: Mobile server, Mobile/Camera Client, and Camera server general flow chart.

distance period. In case a new command is not received by the time the robot finishes the distance period, the robot will stop moving and wait for the next command.

Camera Server: This server is much simpler than the motion server, its job is to simply interface to the camera. This program provides the ability to control the on-board camera using Visca commands. The operator sends pan, tilt and zoom commands, and the server relays these to the camera. This gives the operator the ability to change the field of view to allow close coupling to the environment. When the client sends camera commands (pan, tilt and zoom), the camera server sends these commands to the camera through the serial port and sends back to the client a confirmation of the request.

Mobile/Camera Client : This is the program that runs on the local machine and communicates with both servers and the joystick. The client sends commands to either of the servers based on one of the buttons, whether it is pressed or not. If the button is not pressed the joystick position will be translated into velocity commands and sent to the mobile server. On the other hand, if the button is pressed, the position of the joystick is translated into pan and tilt commands for the camera and two other buttons specify whether zoom in or out is requested. In addition, based on which state the client is in, it will either send the force command back to the joystick once feedback is received or just wait for acknowledgment. The communication with the joystick is done with DirectX technology, and with the servers it is done over the Internet.

In addition, VIC was used to transmit video feedback from the on-board and the overhead cameras to the operator.

7.4.2 Experimental Results

Several experiments were done with this system, where two setups were used for experimentation. One is where the robot is controlled over the same LAN, and the other is where the robot was controlled over the Internet either from Hong Kong or from Japan. The event-based reference was taken as either the control sequence number or the distance the robot traveled.

Teleoperation Over the Same LAN

There are three scenarios for the control over the same network; first case is normal operation, second case is simulated 2 seconds of round trip delay and third case is simulated random time delay. For all these cases, the event-based reference is taken to be the control sequence number.

First, to test the stability of the system, it was operated in a *friendly* environment

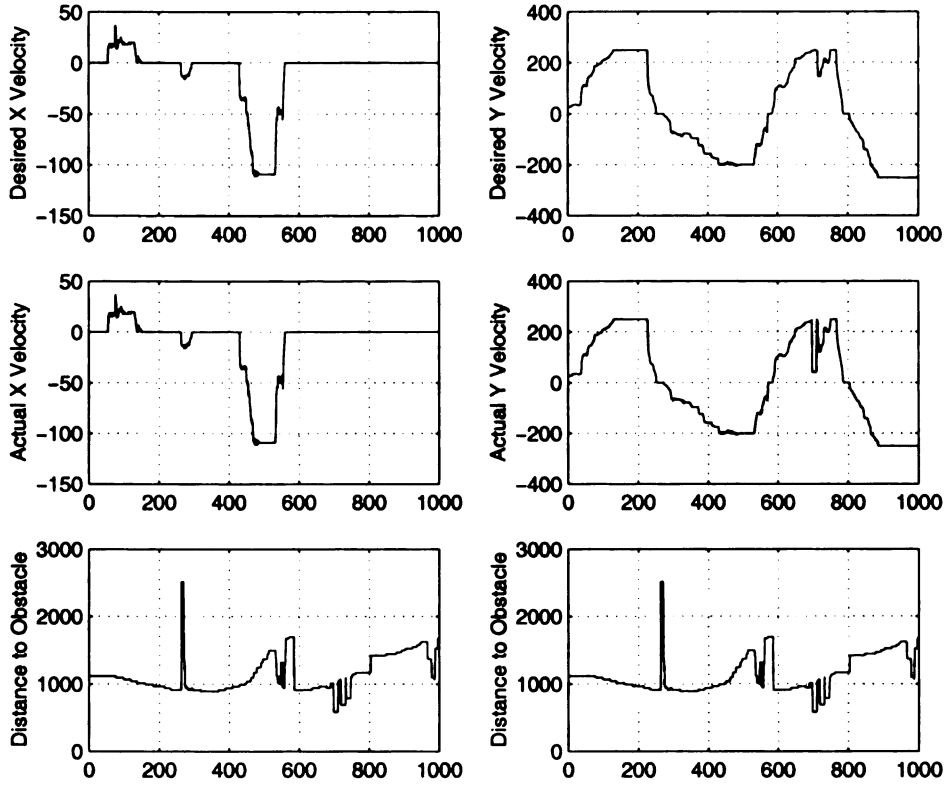

Figure 7.18: The behavior of the system under normal operation with relatively far obstacles.

with relatively far obstacles and the velocity tracking was examined. In this experiment, the operator tries to keep away from obstacles (about 1 meter away), but with no other constraints. Meaning there is no predefined path and the operator is free to move around randomly. Figure 7.18 shows a plot of the operation of the robot with most of the obstacles being more than 1 meter away. The first row shows the desired x and y velocities, the second row shows the actual x and y velocities, and the last row gives the distance detected at each event-based reference. All of these plots are with respect to the event-based reference and not time. It is clear that close tracking of the desired velocities is achieved, except when the robot gets to obstacles which are closer than a meter. So when no obstacles are found the system is showing close tracking performance and is stable. In addition, the system is event-synchronized, since the force rendered at a certain reference corresponds to the distance detected

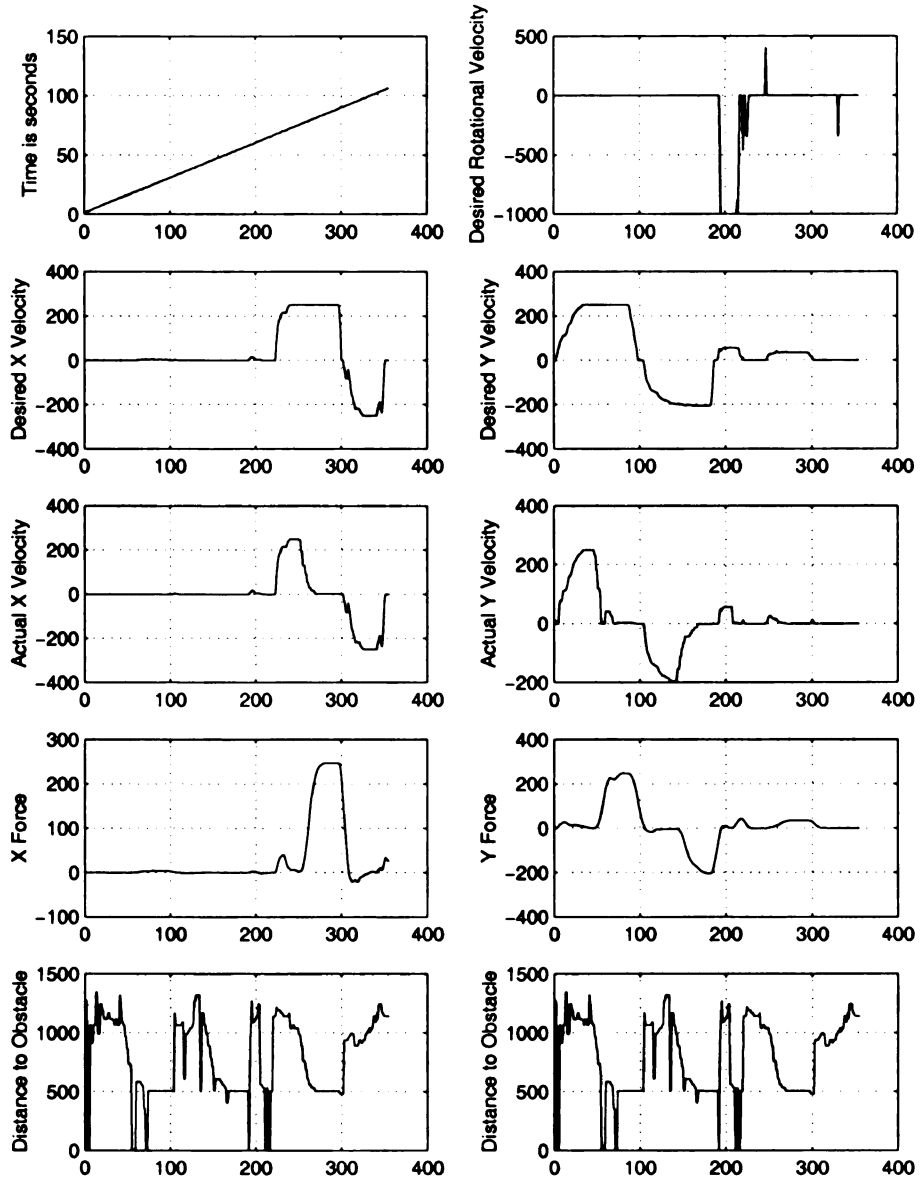

Figure 7.19: The behavior of the system under normal operation.

at that same reference value.

When the environment becomes more *hostile*, that is having several close objects, the behavior of the system changes since now obstacle avoidance is taking place. In all the following experiments the operator is not constrained, complete freedom is given to move around in any direction or fashion and to get close to obstacle as desired.

To understand the system operation under these conditions consider Figure 7.19, which shows the results for the normal operation case. The first row gives a plot of

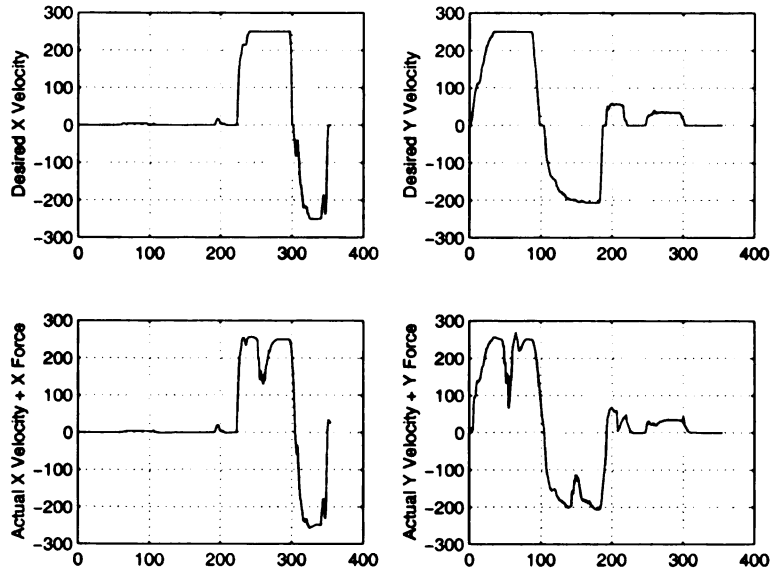

Figure 7.20: Comparison between the desired velocities and the sum of the actual velocities and the forces felt.

time versus s, the event-based reference. It is clear that s is non-decreasing function of time, which is very crucial in ensuring the system stability. The other plot in the first row is the desired rotational velocity. The second row shows the desired velocities in x and y directions, which are called V_m. The third row displays the actual velocities in both directions, which are called V_a. The fourth row gives the forces that are rendered by the joystick in both directions, which are referred to as τ_m. The last row displays the same plot, which is the closest distance detected to obstacles in the environment.

Looking at the actual velocities in the third row, it is clear that it is changing according to the distance detected; for example, if the distance gets closer the actual velocity decreases. When this occurs, the fourth row shows that the force increases as expected. Thus a close object causes the robot tracking error to increase by that conveying to the operator that an obstacle exists. Also note that whenever the robot gets to the critical distance of 0.5 meter the actual velocity becomes zero, which applies to both x and y directions. At the time when the robot gets further away from obstacles the actual velocity starts tracking the desired one again, by that reducing

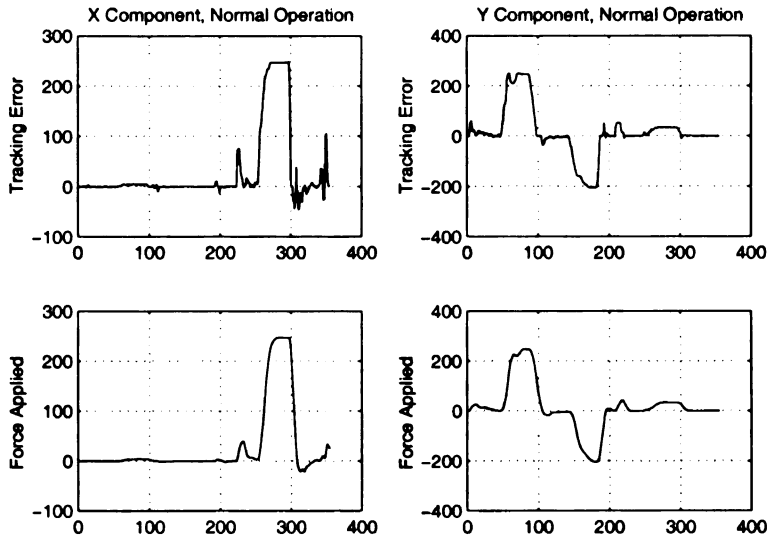

Figure 7.21: Comparison between tracking error and force felt for normal operation.

the tracking error and therefore the force.

Considering the sum of the actual velocity and the force it should be equal to the desired velocity. This is what Figure 7.20 is showing, the first row presents the desired velocities and the second row shows the sum of the forces and the actual velocities. It is clear that the plots are similar although not identical. The difference is due to the fact that the forces rendered are a filtered version of the tracking error as seen in Figure 7.21. This figure shows the tracking error and the actual force felt, which is clearly a low pass filtered version of the error. This is done to smooth the force felt and eliminate abrupt changes. That is why the plots in Figure 7.20 are not identical, since whenever the tracking error changes suddenly the force would change slowly, and that is why the differences in these plots are at the points where the actual velocity is directly reduced to zero, by that causing a big sudden increase in the tracking error. In addition, the system is event-synchronized, since the forces reflect the current distance conditions.

As for the second case, where an additional round trip delay of 2 seconds is induced in the system, the results are shown in Figure 7.22. This figure's layout is similar to the one shown in Figure 7.19. These plots show that the system has a similar behavior

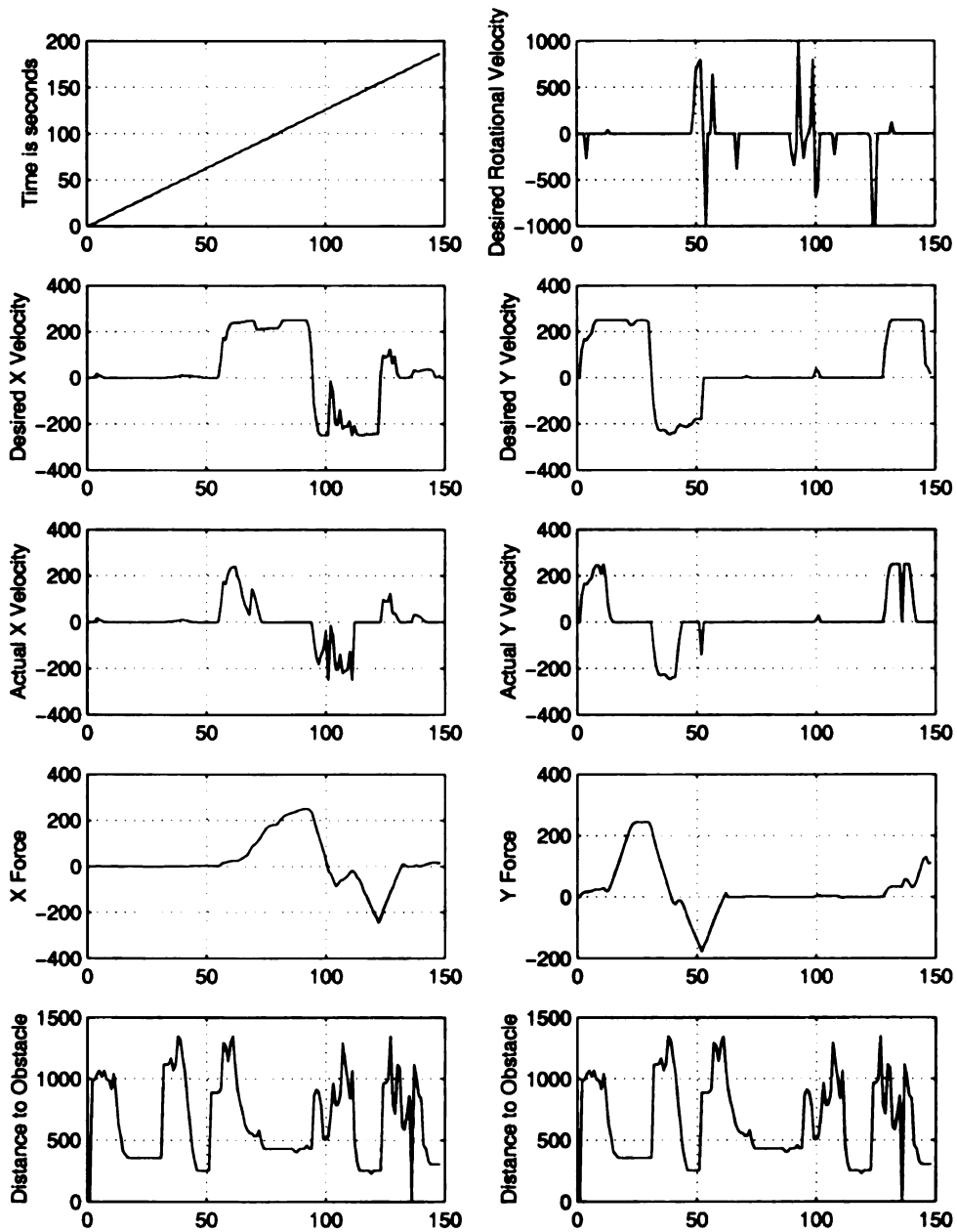

Figure 7.22: The behavior of the system under 2 seconds of round trip delay.

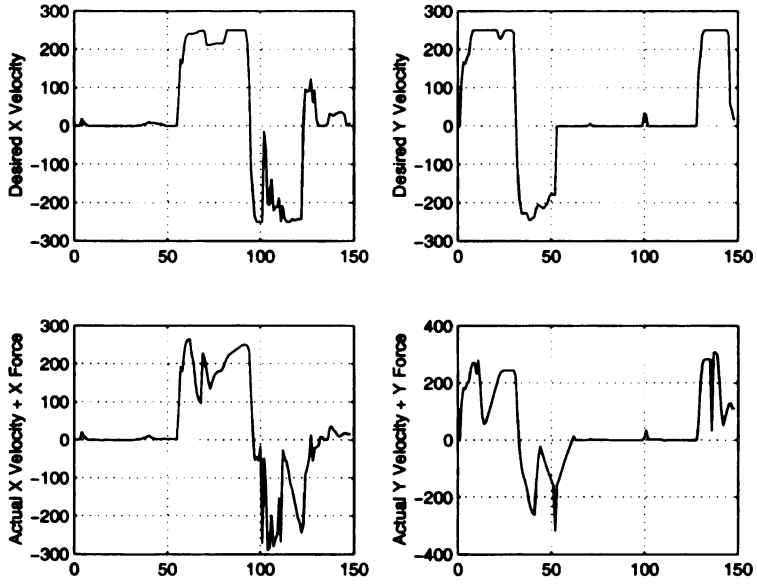

Figure 7.23: Comparison between the desired velocities and the sum of the actual velocities and the forces felt under 2 seconds of round trip delay.

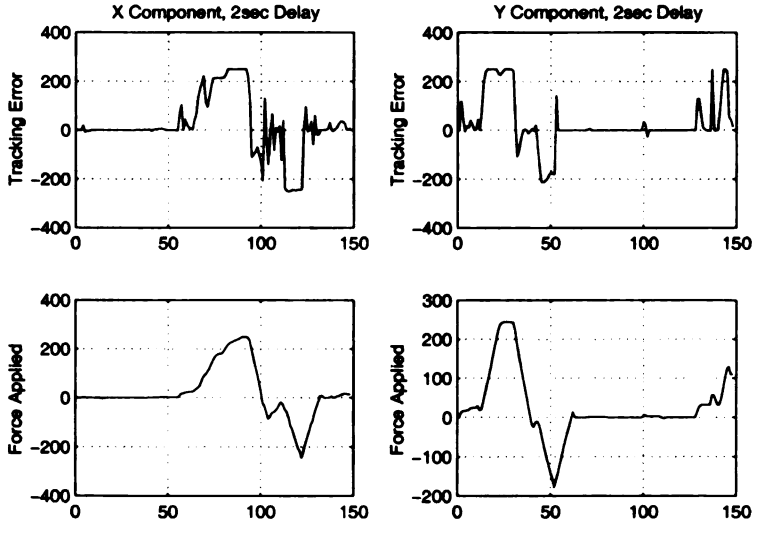

Figure 7.24: Comparison between tracking error and force felt for the 2 seconds delay operation.

to the case where no additional delay exists. The actual velocity tracks the desired one until an obstacle becomes close, then the actual velocity starts deviating from the desired one, by that increasing the tracking error. This increase would increase the force felt. The opposite happens when the robot is getting away from an obstacle, the tracking becomes closer and the error, thus the force, decreases. Again adding the force and actual velocity would give a result similar to the desired velocity as seen in Figure 7.23. The cause of the differences is again the filtering that was mentioned before and seen in Figure 7.24.

The third case includes simulated random time delay, that ranges from 1 to 4 seconds. The results of one such experiment are shown in Figure 7.25, which has a layout similar to Figure 7.19. These results are similar to the results of the previous two cases. The tracking error, that is the force, is a function of the distance detected. The closer the distance the more force is felt and the slower the robot moves, until it comes to a complete stop. Then as the distance increases, the force decreases and the robot follows the commands closer. A plot of the sum of the forces and the actual velocities and of the desired velocities is shown in Figure 7.26. Here too, the errors are due to the filtering illustrated in Figure 7.27.

These results show that the robot is stable and that the operator is event-synchronized with the robot, since the force is felt at the same event-reference the obstacle is detected. More importantly, is that this stability and event-synchronization is unaffected by the time delay, which can be constant or random.

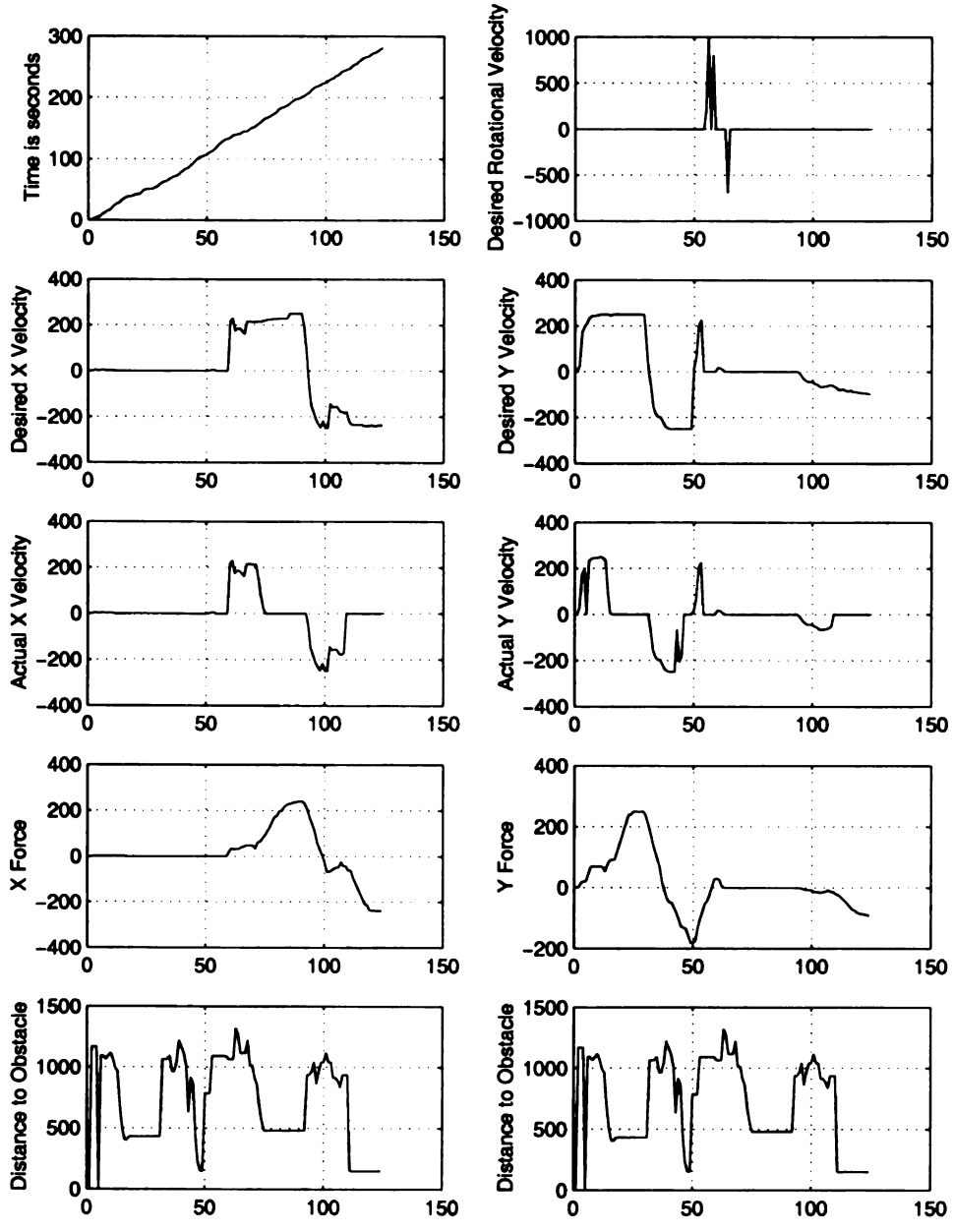

Figure 7.25: The behavior of the system under *random* round trip delay, ranging from 1 to 4 seconds.

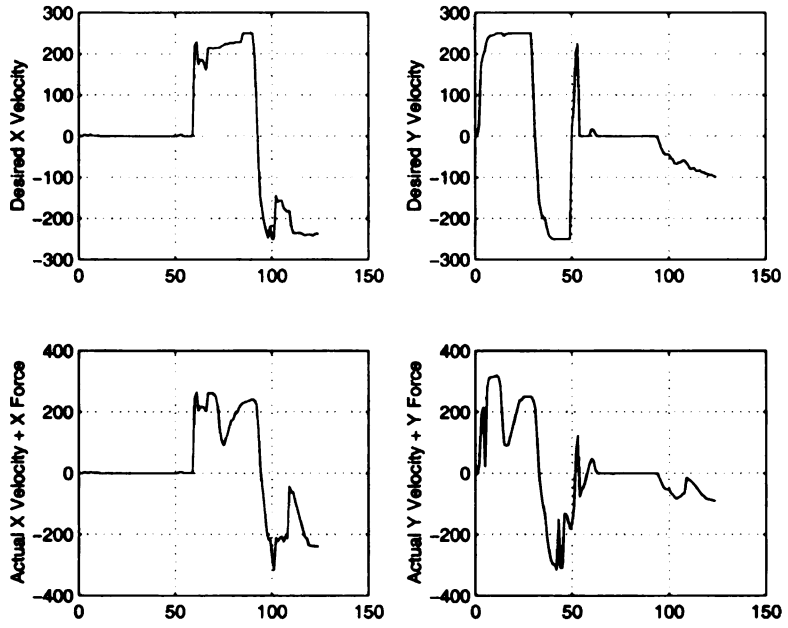

Figure 7.26: Comparison between the desired velocities and the sum of the actual velocities and the forces felt with simulated random delay.

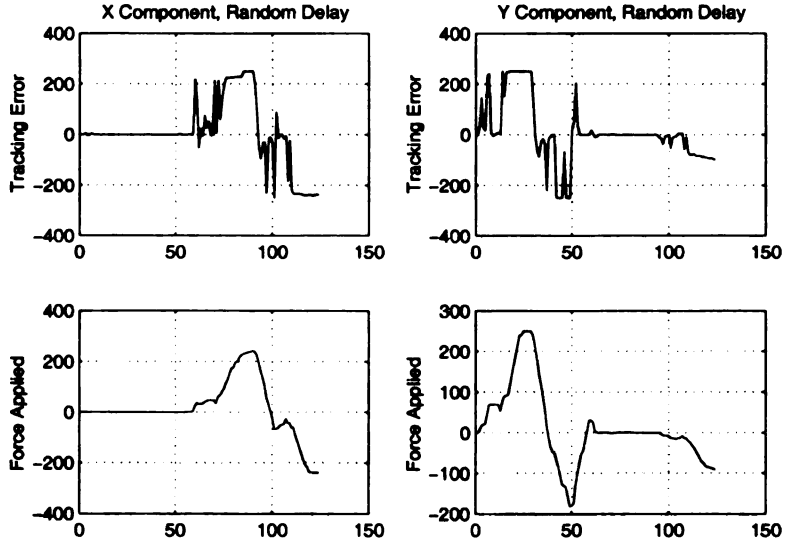

Figure 7.27: Comparison between tracking error and force felt for random delay operation.

Teleoperation from Hong Kong

As for the experiments done with the *Robot Control Lab* in Hong Kong, the control frequency[3] experienced was between 1.4 and 1.5 control cycles/seconds. Some of the results of these experiments are shown in Figure 7.28 and Figure 7.29. These plots are similar in layout to the ones in Figure 7.19. In all of the following experiments, the operator is not constrained, complete freedom is given to move around in any direction or fashion and is allowed to get close to obstacles as desired.

The first plot in the first row is time since the beginning of the connection versus the event-based reference, s, which is taken to be the control sequence number. It is clear that s is a non-decreasing function of time. The second plot in the first row is the desired velocity in the θ direction. The seconds row shows desired velocities in the x and y directions. Under these, given are the actual velocities and forces in the x and y directions. The last row gives the distances to the closest obstacles detected.

Observing these figures, it is clear that the forces increase as the robot gets closer to objects and decreases as it gets further. Meanwhile, the opposite happens to the actual velocity, where it becomes smaller than the desired one as the robot approaches an object, and starts tracking the desired velocity as the robot pulls away, which reflects the stability of the system. In addition, the actual velocity and the force rendered respond to the distance detected at almost the same event reference as this distance, which shows the event-synchronization between the operator and the robot.

As discussed before, the sum of the actual velocity and the force should approach the desired velocity. This is shown in Figure 7.30, where the first and third row show the desired velocity and the second and fourth row show the sum of the force and the actual velocity. As expected, the two are not identical although they are very close. This is a result of the force being a low pass filtered version of the velocity

[3]Control frequency is the number of times per second that the control commands are sent and the feedback received, this gives a measure of events occurring per second.

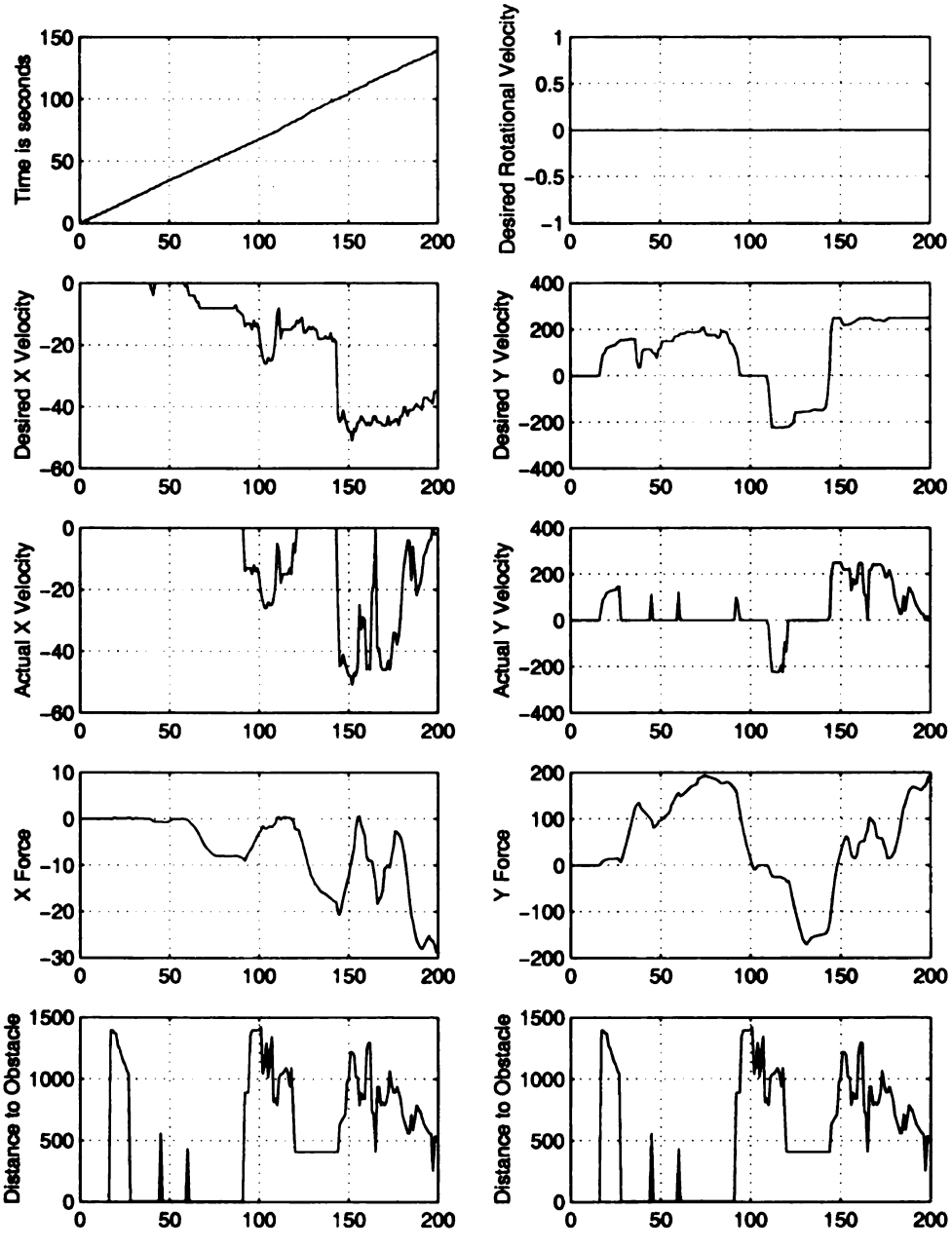

Figure 7.28: The behavior of the system during the control from Hong Kong.

tracking errors. This filtering smoothes the forces as seen in Figure 7.31, which gives the tracking error in the first and third row and the force rendered in the second and fourth row.

To test the sensitivity of the system to the time of day or the network load and to test for event-transparency, the experiments were repeated several times on different days. Figure 7.32, which has a layout similar to Figure 7.28, gives a set of data

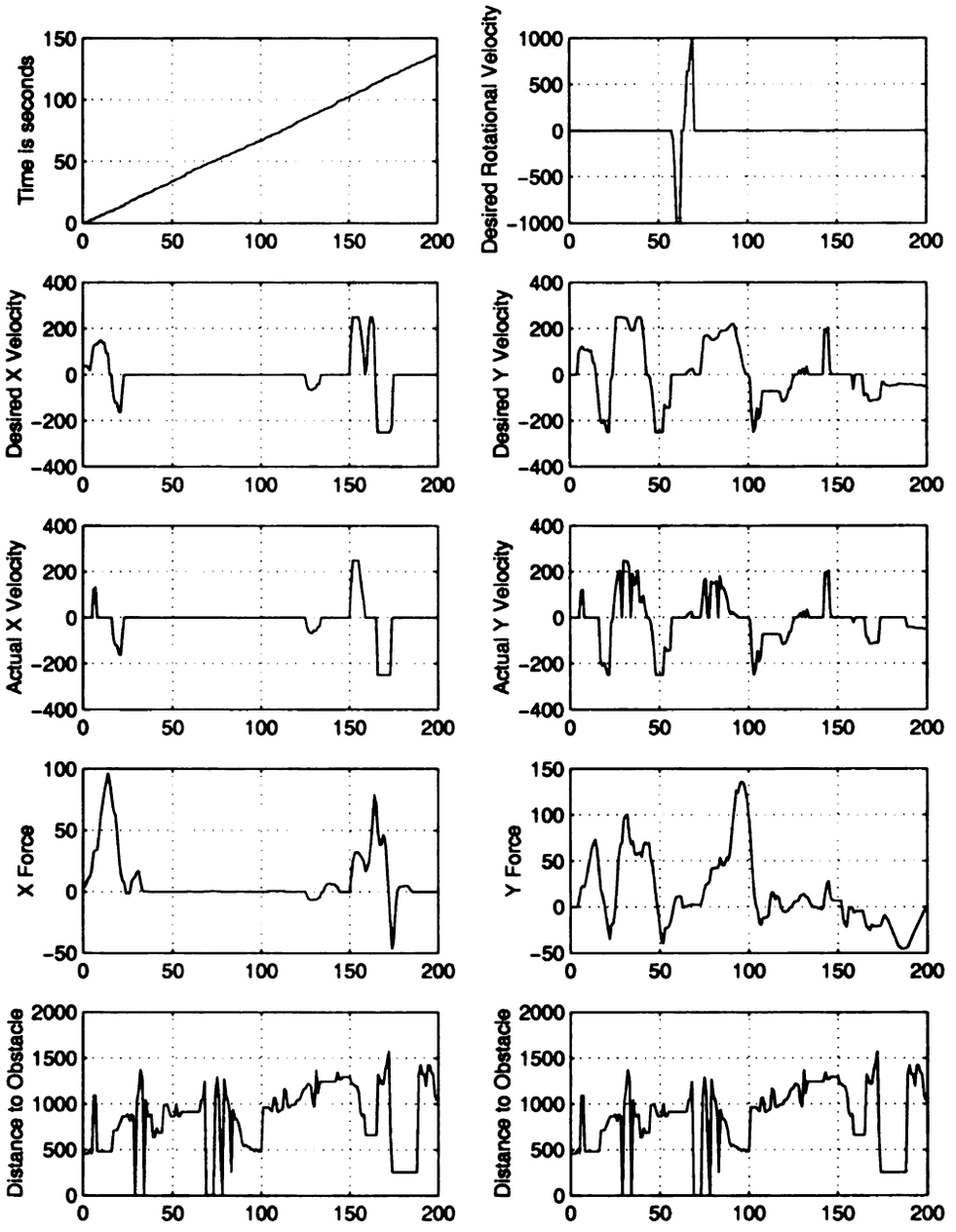

Figure 7.29: The behavior of the system during the control from Hong Kong.

collected at a different day. In this figure the same tracking and obstacle avoidance behavior as in the previous cases is observed. As for Figure 7.33, the similarity between the desired velocity and the sum of the force with the actual velocity is clear. Figure 7.34 simply shows the tracking error and the actual force rendered just to illustrate the filtering process. Since, regardless of the delay, similar control responses were obtained, a consistency is clear. This consistency is nothing but event-

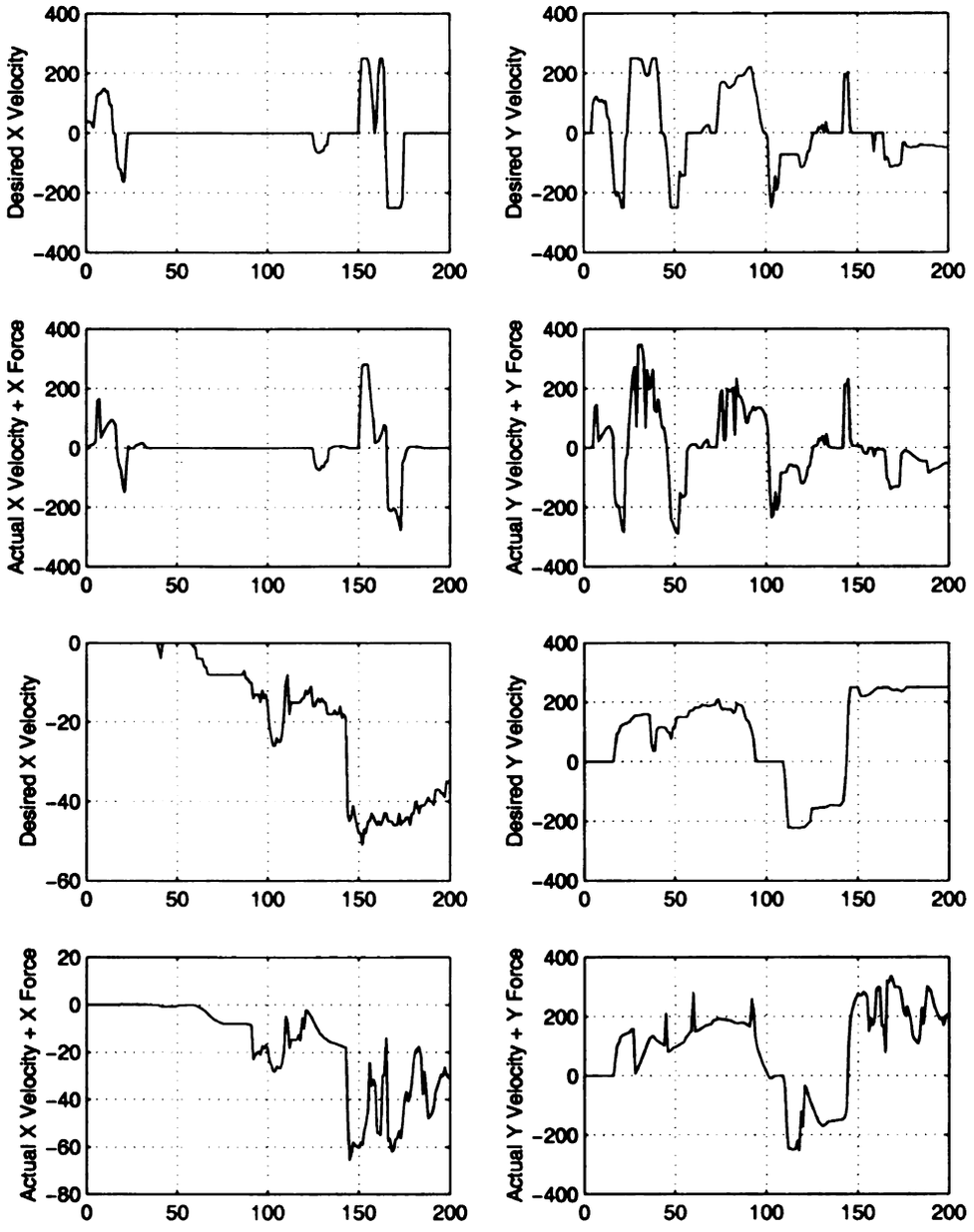

Figure 7.30: Comparison between the desired velocities and the sum of the actual velocities and the forces felt for the two Hong Kong experiments.

transparency.

Another design of this system was the one where the event-based reference was taken to be the distance the robot traveled. The results of teleoperating such a system are shown in Figure 7.35, which has a layout similar to Figure 7.28. It is clear that the performance of the system is similar to the previous cases in terms of stability

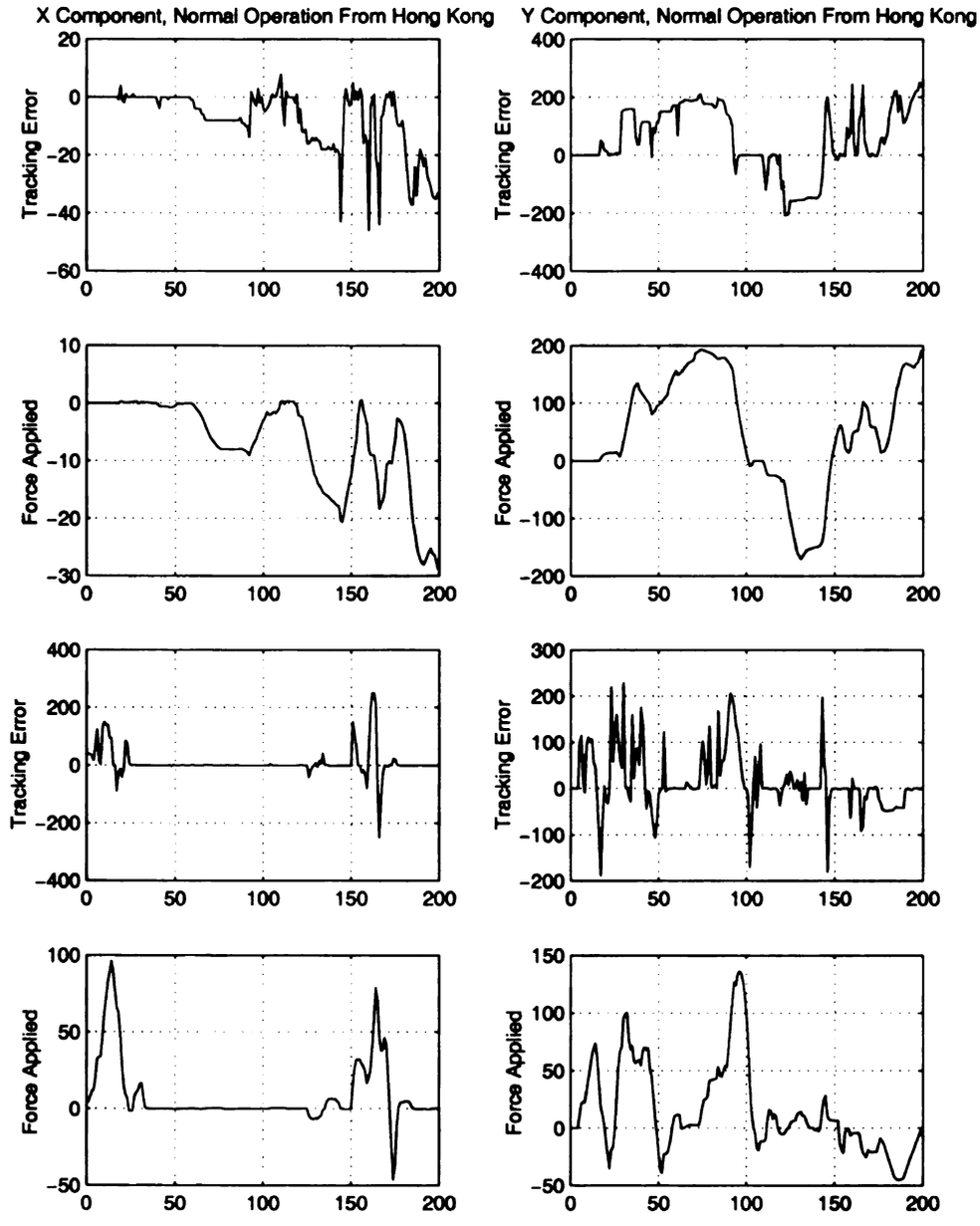

Figure 7.31: Comparison between tracking error and force felt for the two Hong Kong experiments.

and event-synchronization. As for Figure 7.36, it shows the similarity between the desired velocity and the sum of the force with the actual velocity. Figure 7.37 plots the tracking error and the actual force rendered just to illustrate the filtering process.

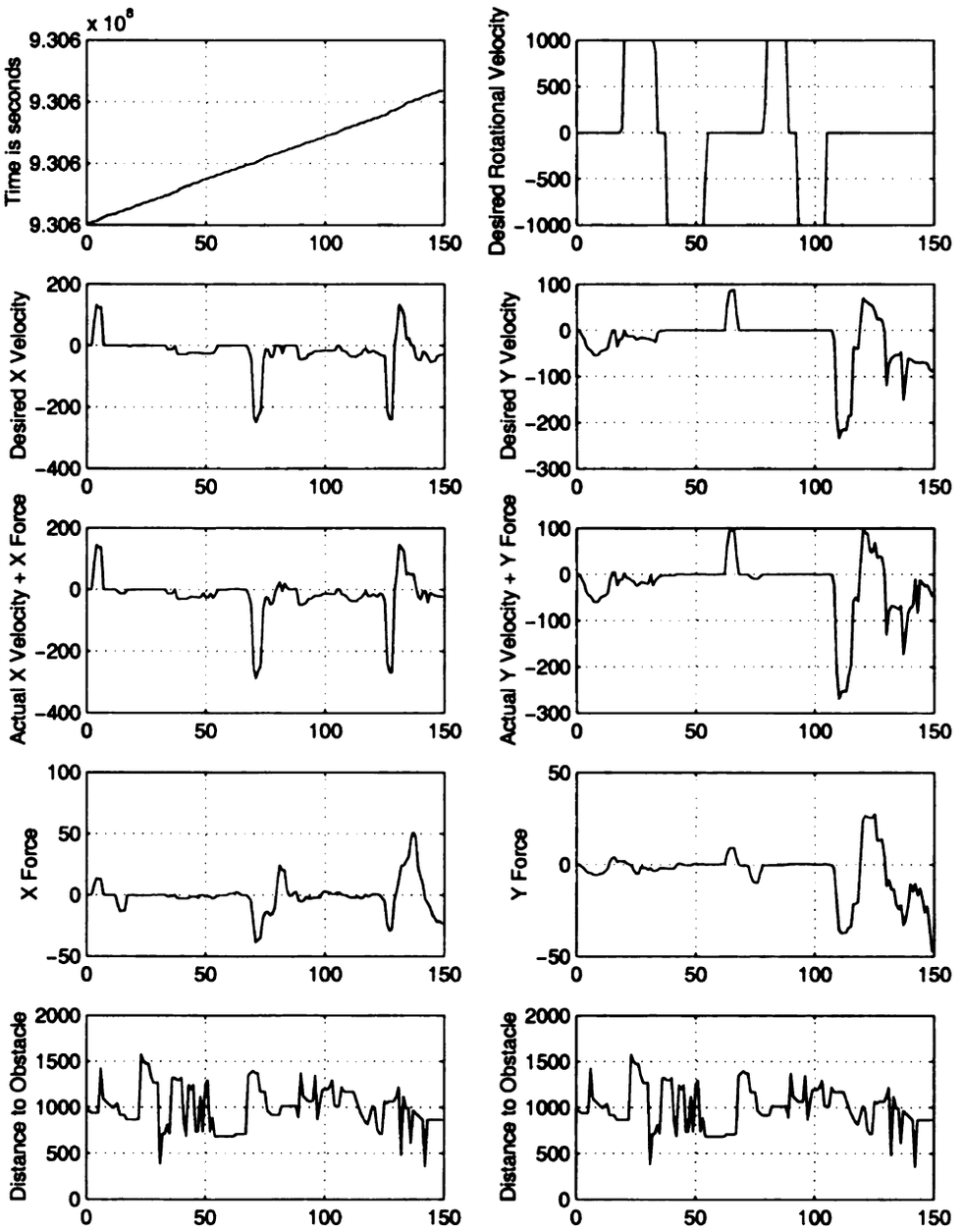

Figure 7.32: The behavior of the system during operation from Hong Kong.

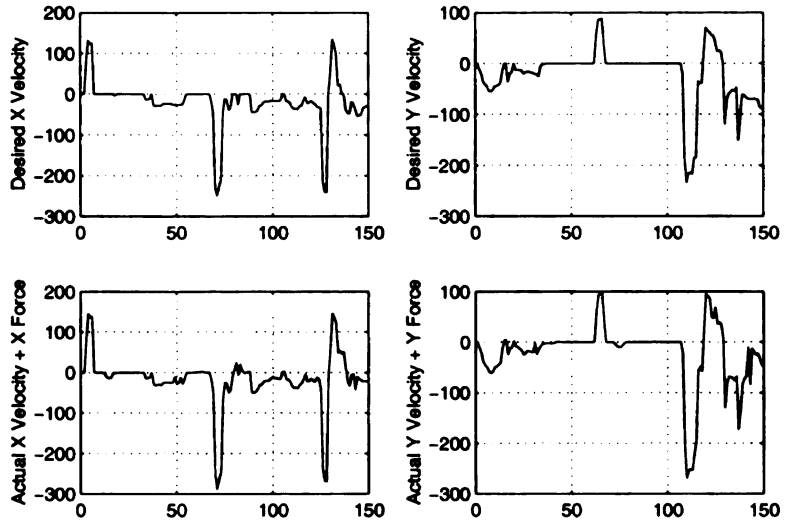

Figure 7.33: Comparison between the desired velocities and the sum of the actual velocities and the forces felt for teleoperation from Hong Kong.

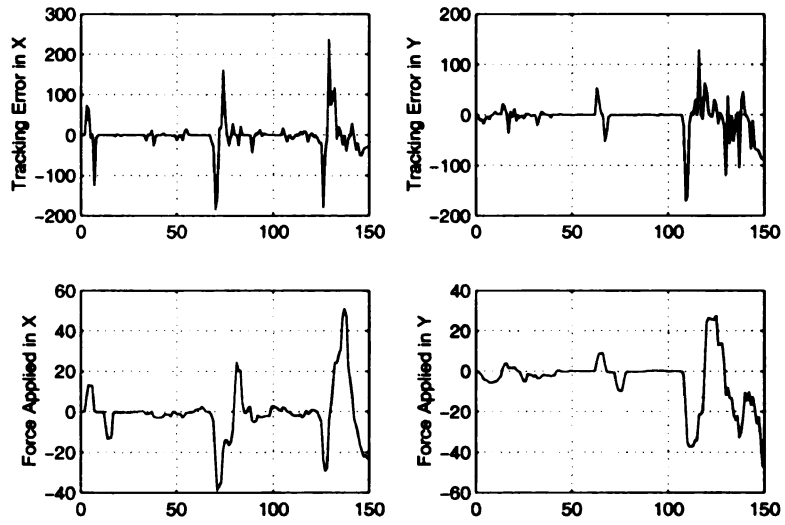

Figure 7.34: Comparison between tracking error and force felt for teleoperation from Hong Kong.

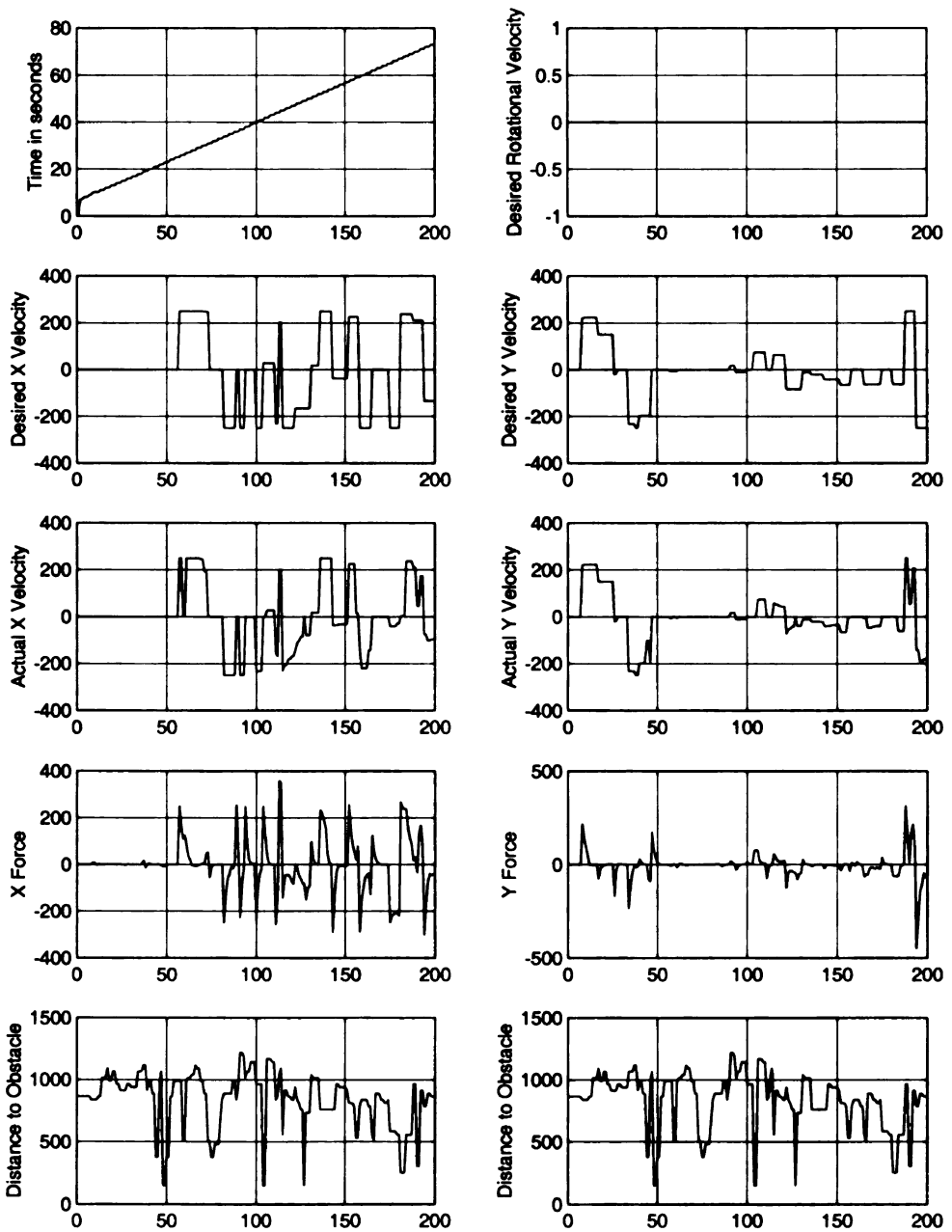

Figure 7.35: The behavior of the system during operation from Hong Kong with event-based reference as distance.

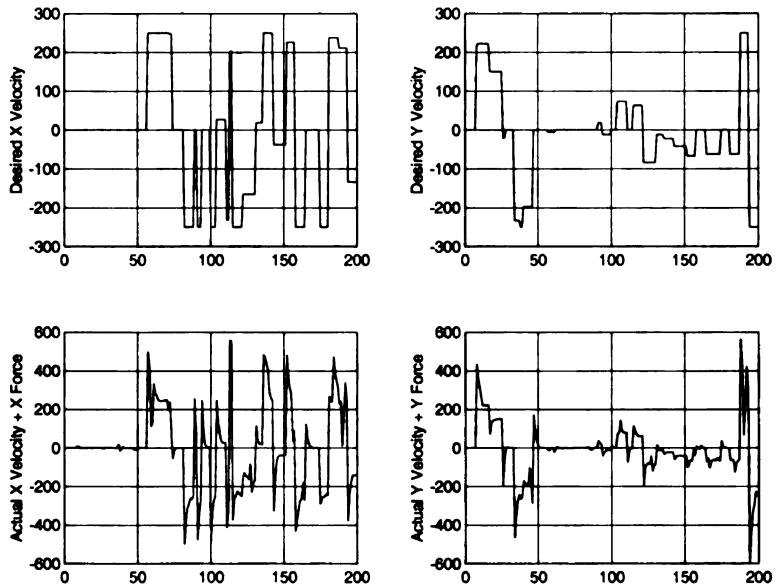

Figure 7.36: Comparison between the desired velocities and the sum of the actual velocities and the forces felt with event-based reference as distance.

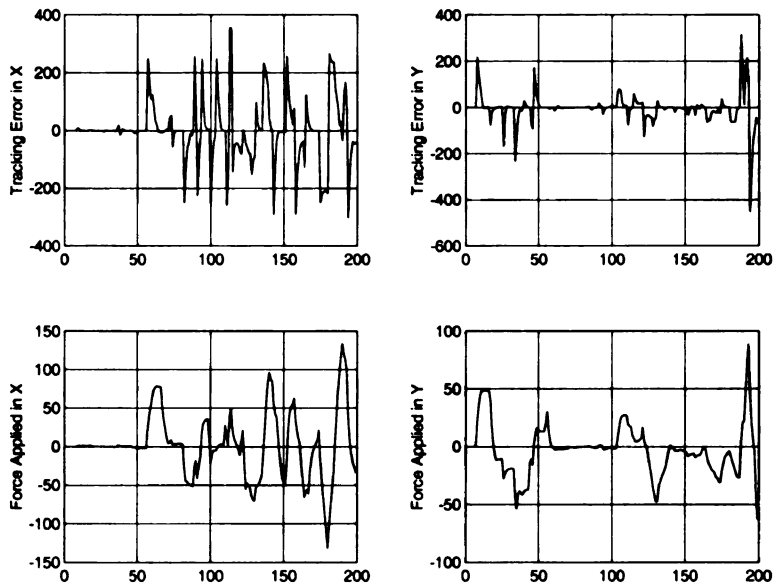

Figure 7.37: Comparison between tracking error and force felt for teleoperation from Hong Kong with event-based reference as distance.

Teleoperation from Japan

The same mobile robot system was teleoperated from Japan. In this case the event-based reference was taken to be the control sequence number. The results of this experiment are shown in Figure 7.38, which has a layout similar to Figure 7.28.

As seen, the performance is similar to all the previous cases. The robot tracks the desired velocity as long as there are no close obstacles, once obstacles are detected the robot slows down and the force rendered increases. This shows that the system is stable. In addition, the response of the system is very fast with respect to the event-based reference, which reflects event-synchronization.

Figure 7.39, which is similar to Figure 7.36, shows that the sum of force and the actual velocity is very close to the desired velocity. As for Figure 7.40, it gives the tracking error in the first row and the force rendered in the second row, where it is clear that the force is simply a filtered version of the error.

All these experiments done with the mobile robot reveal several facts regarding the performance of event-based control and planning. It is clear that the system is stable under random delay conditions and under different choices of the event-based reference. Also the system is event-synchronized, since the force rendered is a reflection of the most recent state of the robot. In addition, the system is exhibiting similar performance between the no delay case, where it was operated over the local network, and the random delay case, whether simulated or real. Therefore, the system is event-transparent despite random delay. Note that this performance was obtained regardless of the operator, the environment or the delay.

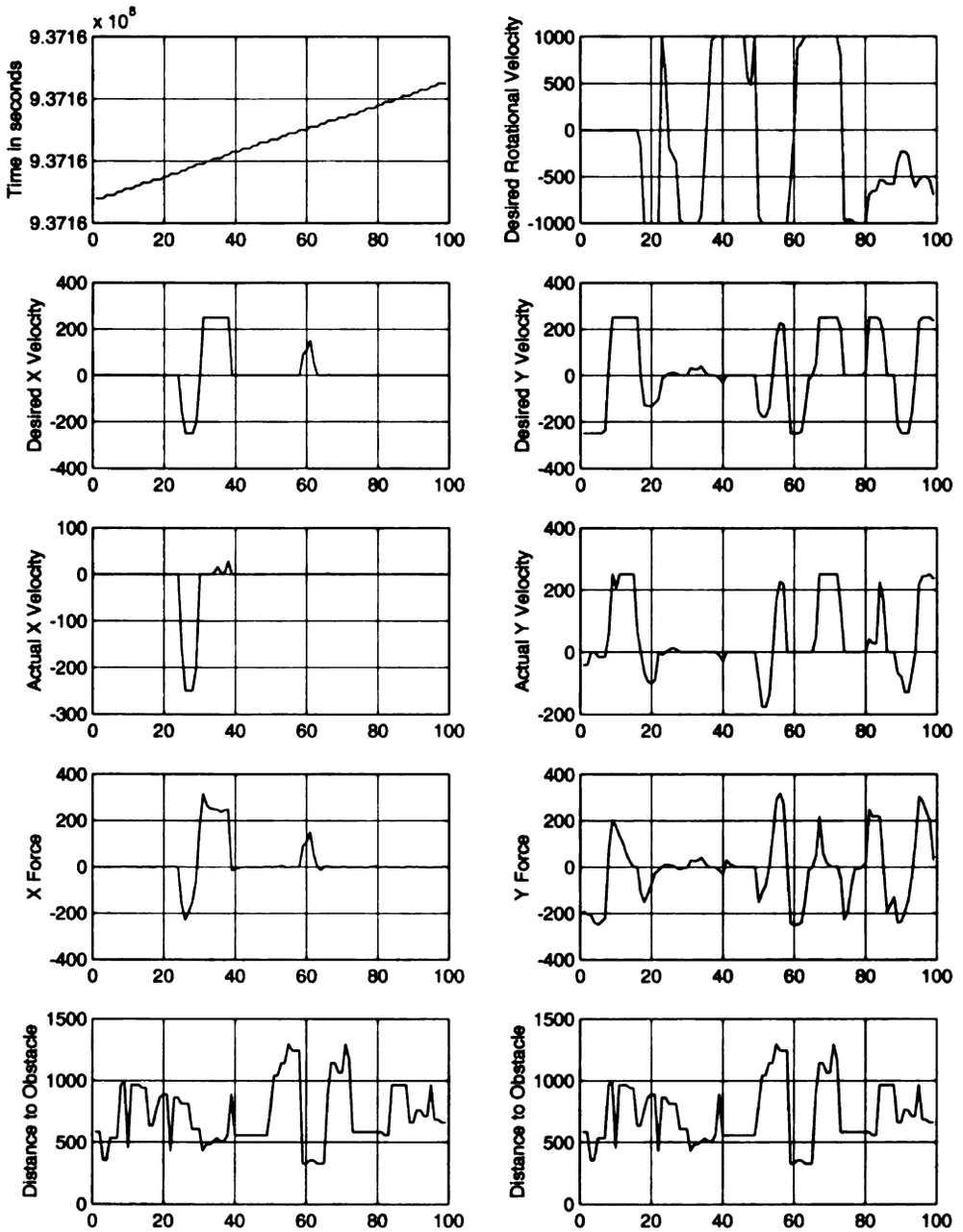

Figure 7.38: The behavior of the system during operation from Japan.

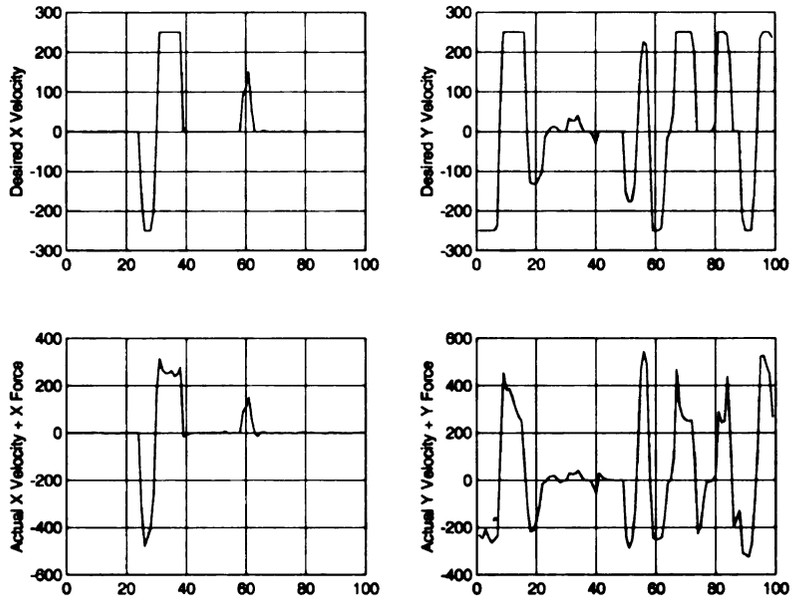

Figure 7.39: Comparison between the desired velocities and the sum of the actual velocities and the forces felt for teleoperation from Japan.

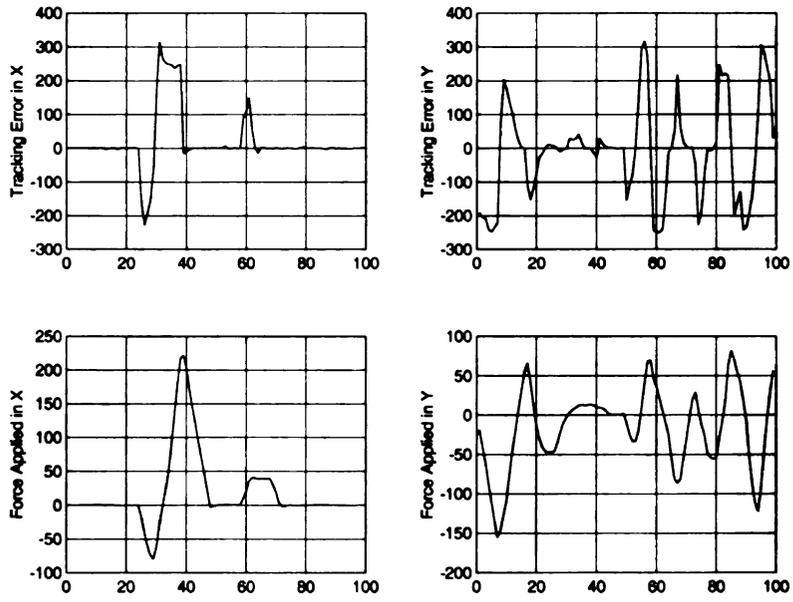

Figure 7.40: Comparison between tracking error and force felt for teleoperation from Japan.

7.5 Mobile Manipulator: Velocity Control

The capability of mobile manipulation is a key to many robotic applications. A mobile manipulator generally consists of a mobile platform and a robot arm. The research on mobile manipulators focuses on motion planning, modeling and control, and force control. Although teleoperation of mobile robots and manipulators have been discussed in the literature; however, the teleoperation of mobile manipulator is relatively new and more difficult. The general form of a mobile manipulator teleoperation system is depicted in Figure 7.41. In this system the operator sends velocity commands and the remote robot feeds back force and video information. This section presents the implementation of an event-based mobile manipulator teleoperation system along with experimental results.

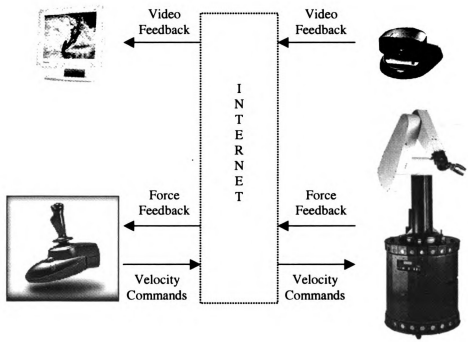

Figure 7.41: General structure of a mobile manipulator teleoperation system.

7.5.1 System Implementation

The implementation discussed will cover both the hardware and the software details of the system developed.

Hardware

The hardware architecture of the system developed is shown in Figure 7.42. The joystick used is a programmable Microsoft SideWinder Force Feedback Pro. with 3 degrees of freedom. The mobile robot is a Nomadic XR4000 and the manipulator is a PUMA560. In addition, other components are used similar to the ones discussed in the general case.

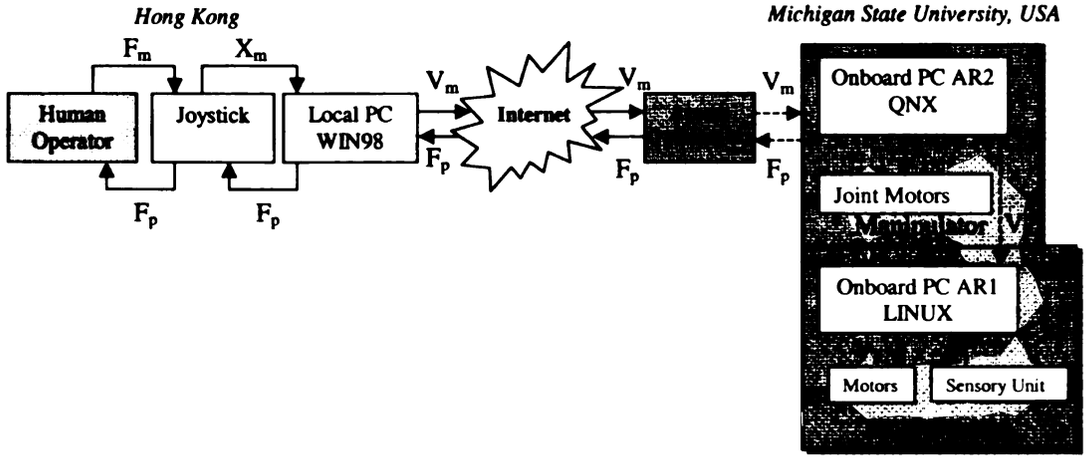

Figure 7.42: Hardware architecture of a mobile manipulator teleoperation system.

Software

The software developed can be divided into three main parts: client, manipulator (PUMA) server and mobile server.

Client: After connecting to the PUMA server, the client sends velocity commands, V_m, to the mobile manipulator and relays force commands back to the joystick once feedback is received. Communication with the joystick is achieved with

MS DirectX technology, and that with the server over the Internet using TCP.

Puma Server: The puma server acts as a server for the client. Once it receives the velocity command from the client via the wireless ethernet, it figures out what part should be executed by the Puma and what part should be executed by the mobile. Then it sends the torque required to the Puma joint motors. At the same time it forwards the velocity required by the mobile to the mobile server. Communication between the two servers is done using TCP via two Ethernet cards connected to each other on the on-board PCs.

In addition, this server uses the force/torque sensor connected to the gripper of the manipulator to detect forces applied on the arm and forwards these to the client to be played back to the operator. In addition to the communication, the Puma server is also responsible for the control of the robot hardware.

Mobile Server: This server receives velocity command from the Puma server and sends these commands to the mobile motors. At the same time it achieves obstacle avoidance by checking the sensors (sonar and laser) and stopping in case an obstacle is detected closer than 0.5 meter.

In addition, the operator receives video feedback from the environment using VIC (Video Conferencing Tool). VIC was used to transmit video feedback from the on-board and the overhead camera to the operator.

7.5.2 Experimental Results

Concerning the experiments done with this implementation, they are divided into two types. The first type is where an operator controls the mobile manipulator locally and the other type is where an operator in Hong Kong controls the robot via the Internet. Since both experiments gave similar results and since the second scenario is more interesting, included here are only results from the testing done with Hong Kong.

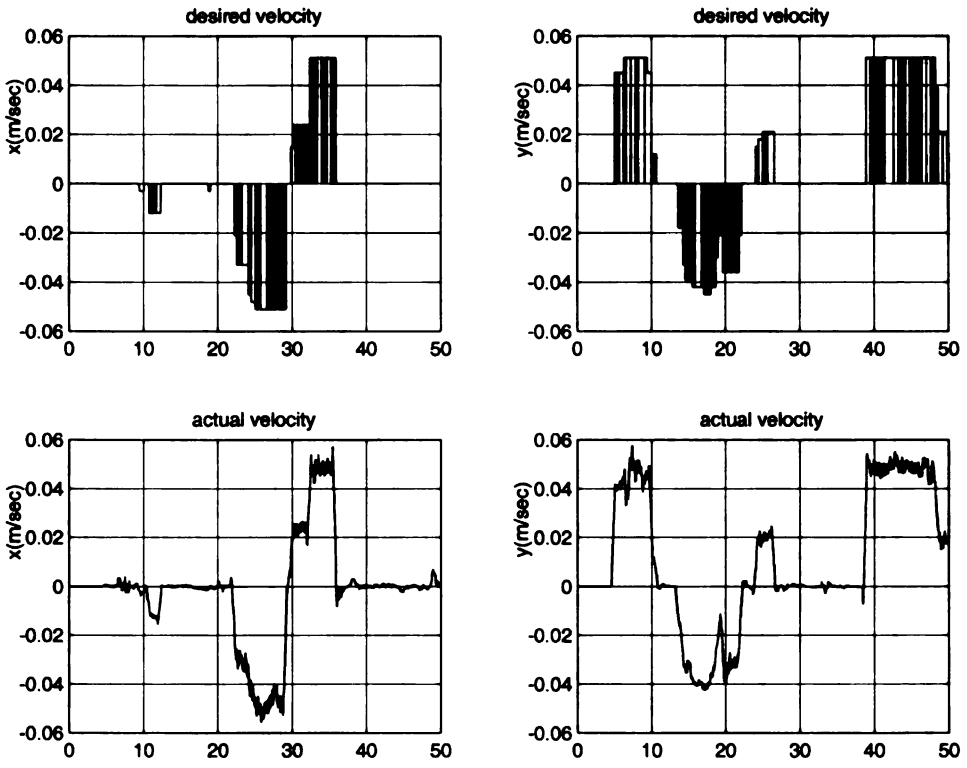

Figure 7.43: Velocity performance of the mobile manipulator teleoperation system.

The experimental procedure had the operator in Hong Kong move the robot in x and y directions randomly. Meanwhile, a person subjects the robot gripper to forces in the x and y directions randomly. The event-based reference was taken to be the control sequence number. The average frequency of communication was $3.1Hz$, which implies that on average there were $3.1events/sec$.

The main interest was in how close the actual velocity of the robot tip was following the desired velocity specified by the operator, and in how close was the force felt by the operator close to the one detected by the force/torque sensor. In other words, how were the operator's intentions executed by the mobile manipulator and how the environment was felt by the operator, despite time delay and time delay variance.

The results are shown in Figure 7.43 and Figure 7.44. In Figure 7.43 the first row gives the operator's desired velocity in the x and y directions with respect to the event-based reference. The second row is the actual velocity of the arm tip in

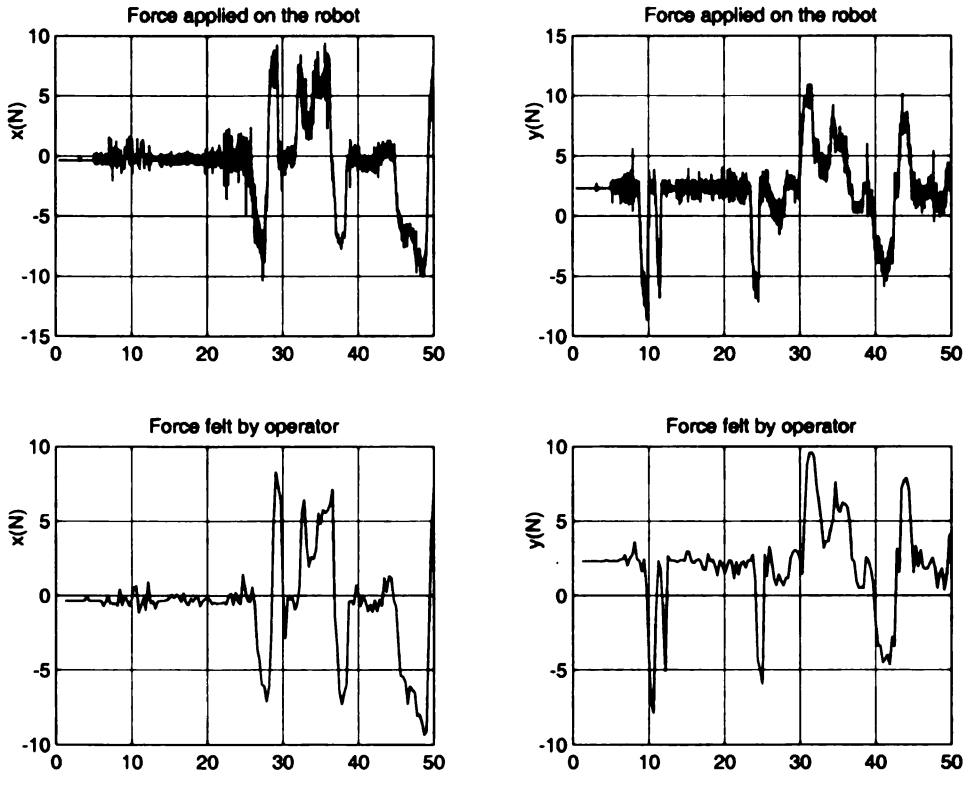

Figure 7.44: Force performance of the mobile manipulator teleoperation system.

the x and y directions with respect to the event-based reference. The first thing to note is that the desired velocity is a step function since it is a sampled version of the continuous motion of the joystick, where the sampling rate is a function of the time delay faced and the advance of the event-based reference. More importantly it is clear that the actually velocity is almost a continuous version of the desired one and it is tracking it almost perfectly, which implies the stability of the system.

As for Figure 7.44, the first row is the force detected by the force/torque sensor and the second row is the force felt by the operator with respect to the event-based reference. As for the force felt, it is clear that it is a step function that is almost tracking the actual one.

It is clear from these results that the system is stable since the actual behavior of the robot is tracking the desired one and the force felt by the operator is very close to the actual one the robot is detecting. In addition, the system is event-synchronized

since the change in the rendered force occurs almost at the same reference as the change in the detected force.

7.6 Mobile Manipulator: Position Control

As discussed in the previous section, mobile manipulators have several usages and advantages. However, using velocity control as shown in the previous section does not allow for the use of all the degrees of freedom of the arm and lacks high accuracy. To achieve this, the Phantom haptic device is used as a master instead of the MS Force Feedback joystick. In this case, the operator supplies position increment commands and the robot feeds back actual force resulting from the interaction with the environment. The general architecture of such a system is shown in Figure 7.45.

This system is experimented using tasks such as picking up and manipulation. The objective from this system is to illustrate the generality of the approach and to show that the system can have control commands other than velocity. This illustrates that the design is independent of the specific commands used. Also this system renders tasks requiring high levels of accuracy feasible.

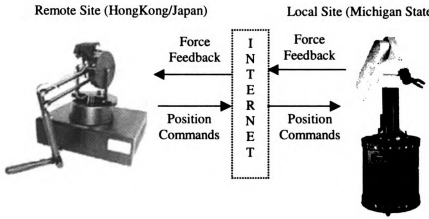

Figure 7.45: The architecture of a supermedia enhanced PUMA manipulator teleoperation system

7.6.1 System Implementation

This section gives the details of the hardware and software utilized and implemented as part of this system.

Hardware

The hardware used is similar to the one shown in Figure 7.42. The only difference from the system discussed in the previous section is the use of a Phantom haptic device instead of the MS Sidewinder Force Feedback Pro joystick.

Software

Similar to the system discussed in the previous section, the software developed in this case can be divided into three main parts: client, manipulator (PUMA) server and mobile server.

Client: After connecting to the PUMA server, the client sends position increment commands to the mobile manipulator and relays force commands back to the Phantom once feedback is received. Also the operator has the capability of opening and closing the gripper.

Puma Server: The puma server acts as a server for the client. Once it receives the position increment commands from the client via the wireless ethernet, it figures out what part should be executed by the Puma and what part should be executed by the mobile. Then it sends the torque required to the Puma joint motors. At the same time it forwards the position required by the mobile to the mobile server. Communication between the two servers is done using TCP via two Ethernet cards connected to each other on the on-board PCs.

In addition, this server uses the force/torque sensor connected to the gripper of the manipulator to detect forces applied on the arm and forwards these to the client

to be rendered to the operator. In addition to the communication, the Puma server is also responsible for the control of the robot hardware.

Mobile Server: This server receives position commands from the Puma server and sends these commands to the mobile motors. At the same time it achieves obstacle avoidance by checking the sensors (sonar and laser) and stopping in case an obstacle is detected closer than 0.5 meter.

In addition, the operator receives video feedback from the environment using VIC (Video Conferencing Tool). VIC was used to transmit video feedback from the on-board and the overhead camera to the operator.

7.6.2 Experimental Results

Several experiments where carried out from Hong Kong via the Internet. The operator commanded the manipulator using the Phantom haptic device, which was used to render force information fed back from the environment. Since the control is event-based there is a need to investigate the integrity of the force signal being sampled with respect to time. That is there is a need to compare the fed back force signal, which is sampled with respect to the event-based reference, with the continuous force sensed with respect to time. For this reason Figure 7.46 gives in the first row the x, y and z components of the continuous force sensed with respect to time and in the second row the x, y and z components of the discrete force fed back with respect to time.

It is clear that the signals are very similar, with the fed back signal shown in the second row being a low pass filtered version of the continuous signal shown in the first row.

In addition, to check the integrity of the received force signal compared to the fed back signal Figure 7.47 is provided. The first row plots the x, y and z components of the force signal fed back and the second row plots the x, y and z components of the

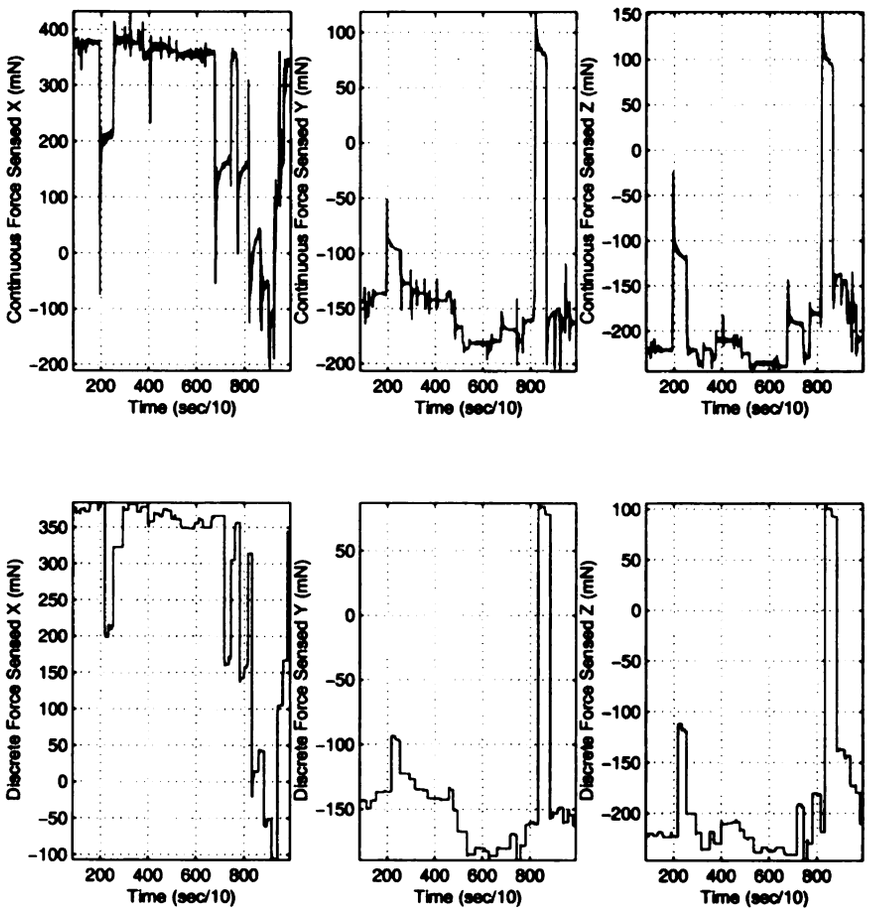

Figure 7.46: Comparison between the continuous force sensed by the robot and the discrete force fed back both with respect to time.

force signal received by the master all of which are with respect to the event-based reference. As seen, except for small errors due to rounding, the two signals are almost identical.

Since the force received is discrete there is a need to smooth it out before rendering. For this end the force planner, which runs at a higher frequency than the communication loop, adjusts the desired force at small increments so that no sudden jerks are felt. This is illustrated in Figure 7.48, which compares the force received with the actual force rendered by the master shown in rows one and two respectively. As expected the actual force rendered is simply a smoothed version of the force received.

Those figures show that the rendered force is almost identical to the received force, which is almost identical to the fed back force, which is almost identical to the actual

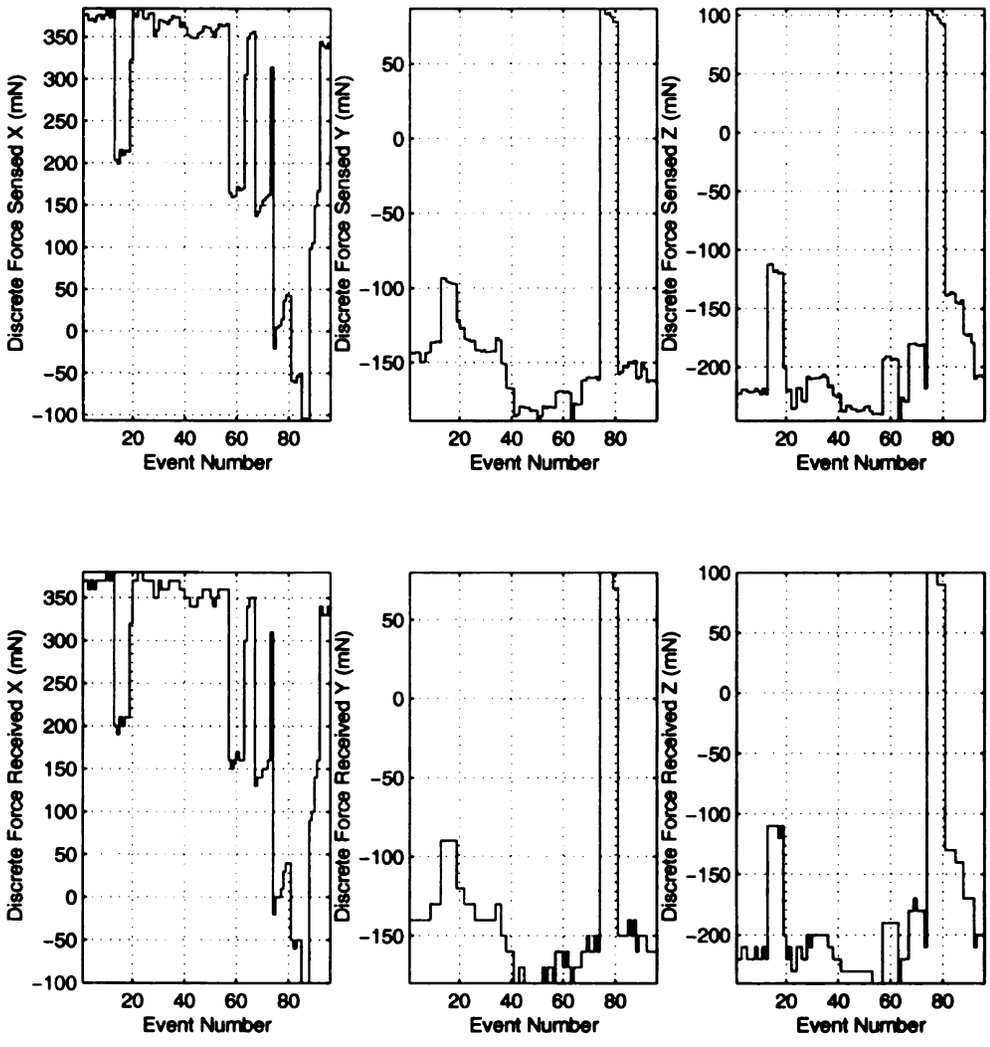

Figure 7.47: Comparison between the force fed back from the robot and the force received by the master both with respect to the event-based reference.

force. Therefore, it is clear that the force sensed by the robot is sampled, transmitted, received and rendered uncorrupted.

As for the manipulator control, the comparison between the desired and actual positions is given in Figure 7.49. In this figure, the first row shows the x, y and z components of the desired position and the second row shows the x, y and z components of the actual position all with respect to the event-based reference. For all the components the signals are very similar with most of the errors being due to rounding and inaccurate position sensing. This shows that the system is stable under event-based planning and control regardless of the delay encountered.

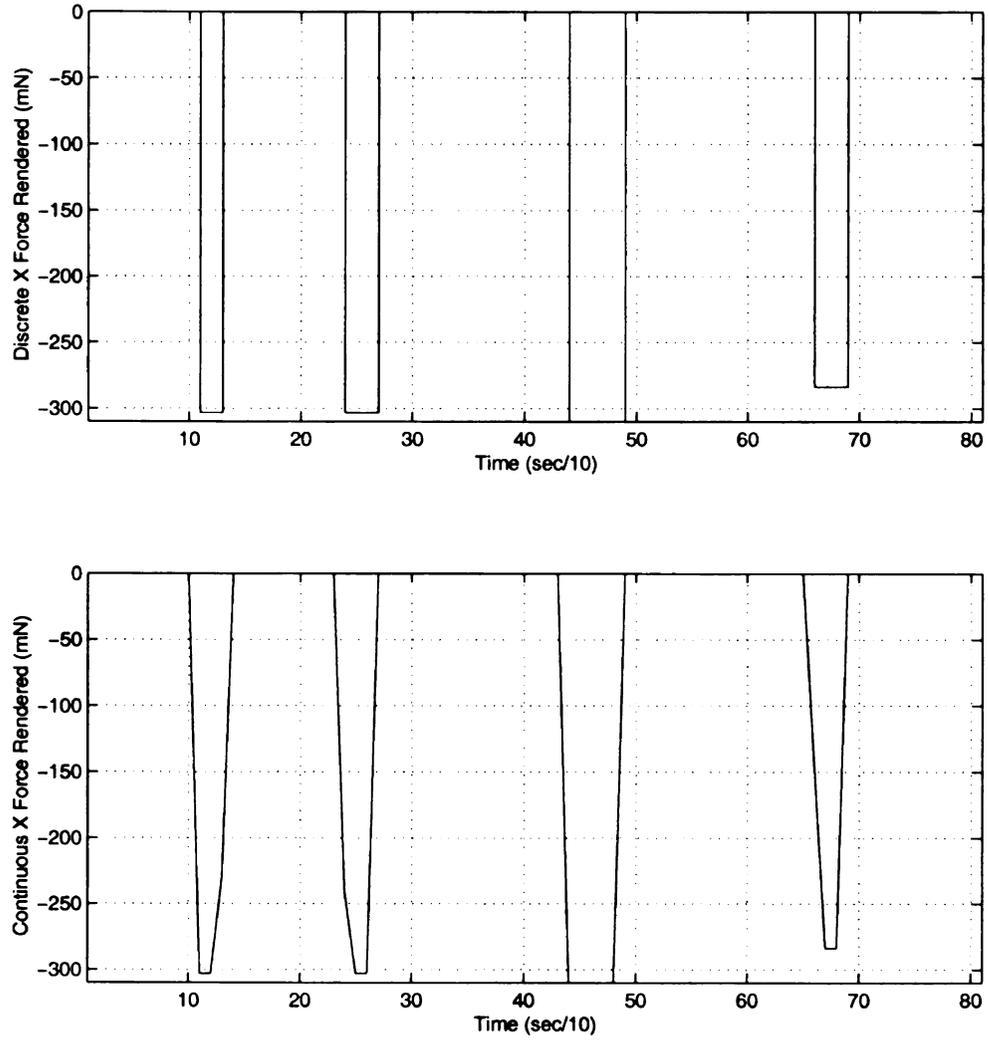

Figure 7.48: Comparison between the discrete force received by the phantom and the actual continuous force rendered by the phantom both with respect to time.

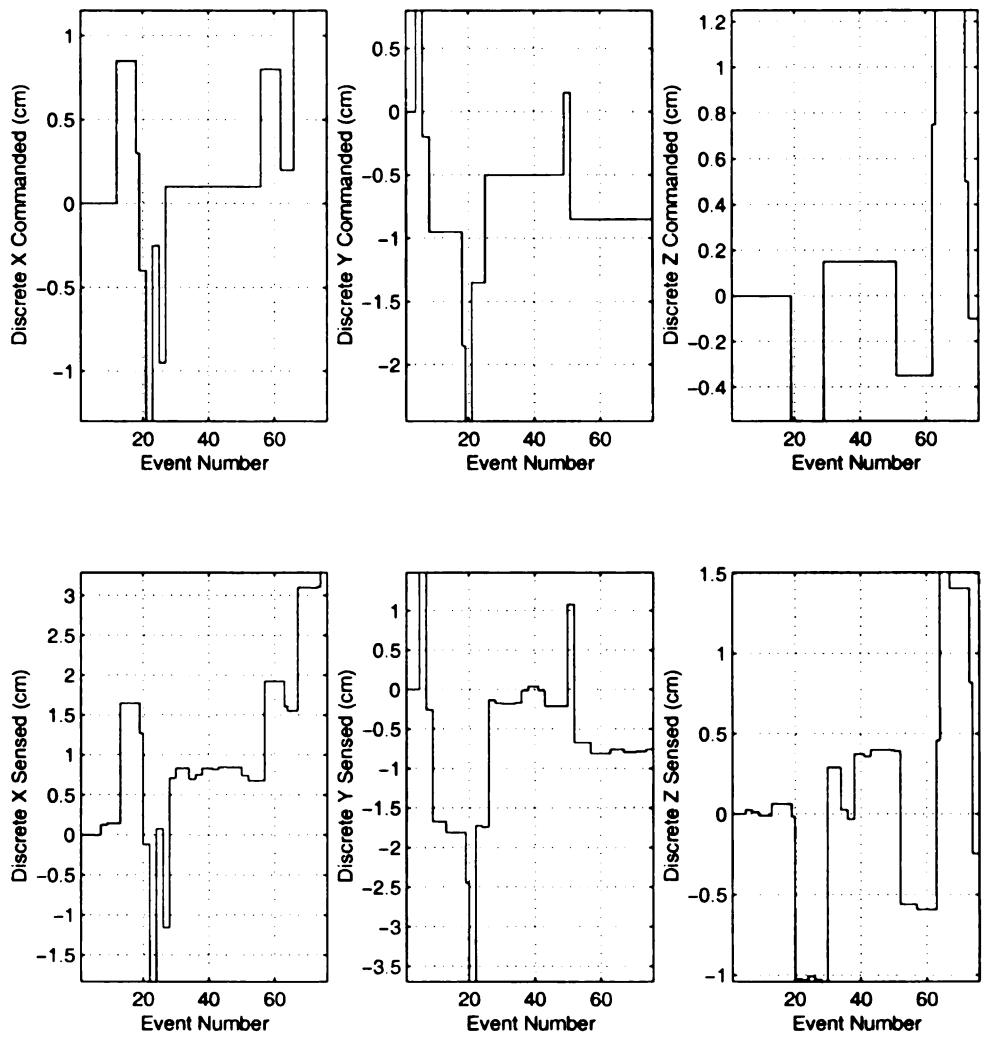

Figure 7.49: Comparison between the desired position and the actual position both with respect to the event-based reference.

7.7 Multi-Site Multi-Operator Tele-Cooperation

Most of the work done relating to cooperation in robotics can be divided into two categories: human-robot and multi-robot. Human-robot cooperation occurs when a human and a machine work simultaneously together to achieve a task, which is impossible for the human or the robot to carry on alone. This approach tries to combine the human intelligence with the robot's physical strength [108] [109]. Multi-robot refers to instances where two or more robots work together to finish an operation that might be difficult for one robot to perform or an operation which, by its nature,

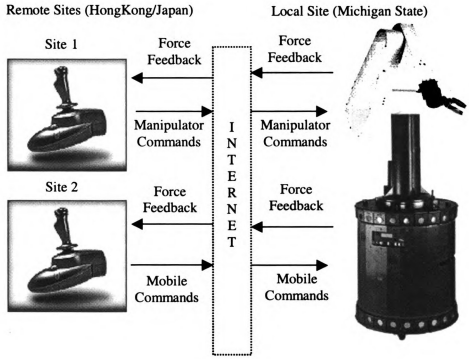

Figure 7.50: The structure of a multi-operator at multi-site collaborative teleoperation system.

might require group work [110]-[112].

Multi-operator at multi-site collaborative teleoperation is a combination of teleoperation and cooperation, where several operators at different remote sites are trying to collaboratively control a robot or several robots to achieve a certain task [113]. The motivation behind tele-cooperation is the existence of tasks that require several different expertise to collaborate concurrently. In cases where these expertise are at different locations, real-time collaborative teleoperation via the Internet is the answer.

A special case of a tele-cooperation system is depicted in Figure 7.50, where two operators are collaborating to bilaterally control a mobile manipulator. This section details the implementation of an event-based multi-operator multi-robot teleoperation system along with experimental results.

7.7.1 System Implementation

Hardware and software details will be covered followed by experimental results.

Hardware

Figure 7.51 presents the hardware architecture of a multi-operator multi-robot collaborative teleoperation system in which one operator controls the mobile base and the other controls the manipulator. The specifications of the different hardware components are similar to the ones discussed in the previous sections.

These different components interact and communicate in the following manner. After each operator connects to a robot, the following sequence of events is repeated until one of the operators disconnects. The operators move the joystick to a certain position which corresponds to a velocity vector. Then the PC of operator2, who is controlling the manipulator, generates the desired velocity vector, V_p, and sends it to the manipulator for execution. Meanwhile, the PC of operator1, who is controlling the mobile base, generates the desired velocity vector, V_m, and sends it to the mobile base for execution. Once the on-board PC (AR2), which is connected to the manipulator, receives V_p it sends this desired velocity to the joint motors controller for execution. Then AR2 waits for the other on-board PC (AR1), which is controlling the mobile base, to be ready to carry out the feedback exchange. In parallel, once AR1 receives V_m it sends this desired velocity to the wheel motor controller for execution. The controller does not execute V_m blindly but engages an obstacle avoidance algorithm. Meanwhile, AR1 waits for AR2 to exchange feedback–AR2 forwards V_p to AR1 and AR1 forwards V_m to AR2. After this exchange is done, AR2 and AR1 feedback V_m and V_p to operator2 and operator1 respectively. This implies that the operators are aware of each other's intentions, which makes collaboration more efficient and safe.

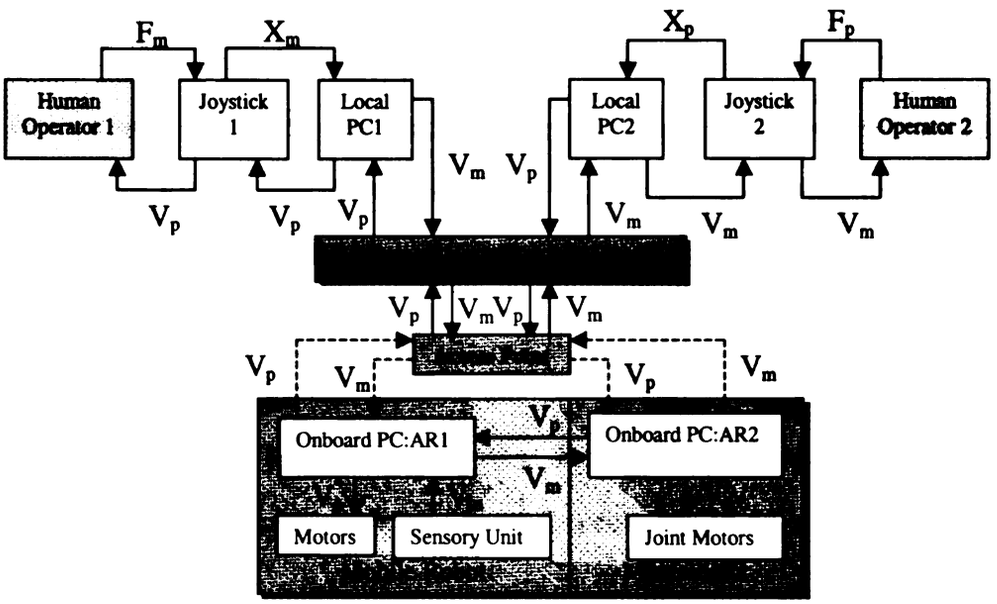

Figure 7.51: Hardware architecture of the multi-operator at multi-site collaborative teleoperation system.

Software

The software developed can be divided into five main parts: mobile server, puma server, puma controller, mobile client and puma client.

Mobile Server: Its service is moving the robot and receiving feedback. However, the server does not execute requests blindly; it first checks the sensors and based on input from them makes a decision according to an obstacle avoidance algorithm. Once the mobile server decides which velocity to set and sends it to the motors, it waits for the puma server to send the feedback V_p. Then it would send V_m to the puma server and V_p to the mobile client.

Puma Server: This server is responsible for receiving the puma velocity commands, V_p, from operator2. Then it forwards these commands to the puma controller. After that the puma server sends V_p to the mobile server and receives V_m from it. Then V_m is forwarded to the puma client.

Puma Controller: This controller receives the desired velocity, V_p, and controls the puma joint motors. The puma controller is designed at task level. A singularity-

free hybrid motion controller is used to avoid the singularities of the robot arm [79].

Mobile Client: This client sends commands, V_m, to the mobile server and relays force commands back to the joystick once feedback is received. Communication with the joystick is achieved with MS DirectX technology, and that with the server over the Internet.

Puma Client: This program sends velocity commands, V_p, to the puma server and relays force commands back to the joystick once feedback is received. Communication with the joystick is achieved with DirectX technology and with the server over the Internet.

Although force feedback is very helpful it does not eliminate the need for visual feedback, which was supplied to the operators from an overhead camera using VIC.

7.7.2 Experimental Results

This section describes two of the experiments that were done, in which the mobile manipulator (Robotics and Automation lab, Michigan State University), operator1 (Robot Control lab, Chinese University of Hong Kong) and operator2 (Nagoya University, Japan) were connected via the Internet. The delay experienced between these sites is random with no specific pattern or model. These two experiments encompass the two most popular modes of collaboration. These modes are master-slave, in which one operator is leading the task, and master-master, in which the two operators are helping each other without having a leader.

In the master-slave operation, one of the operators is requested to follow the force felt. This implies that the slave would eventually track the motion of the master. The results of one such experiment are seen in Figure 7.52, where the operator in Japan is controlling the mobile (slave) and the operator in Hong Kong is operating the puma (master). The results show that the desired velocity of the mobile is tracking that of the puma in real-time, which reflects stability. The top row of Figure 7.52 shows a

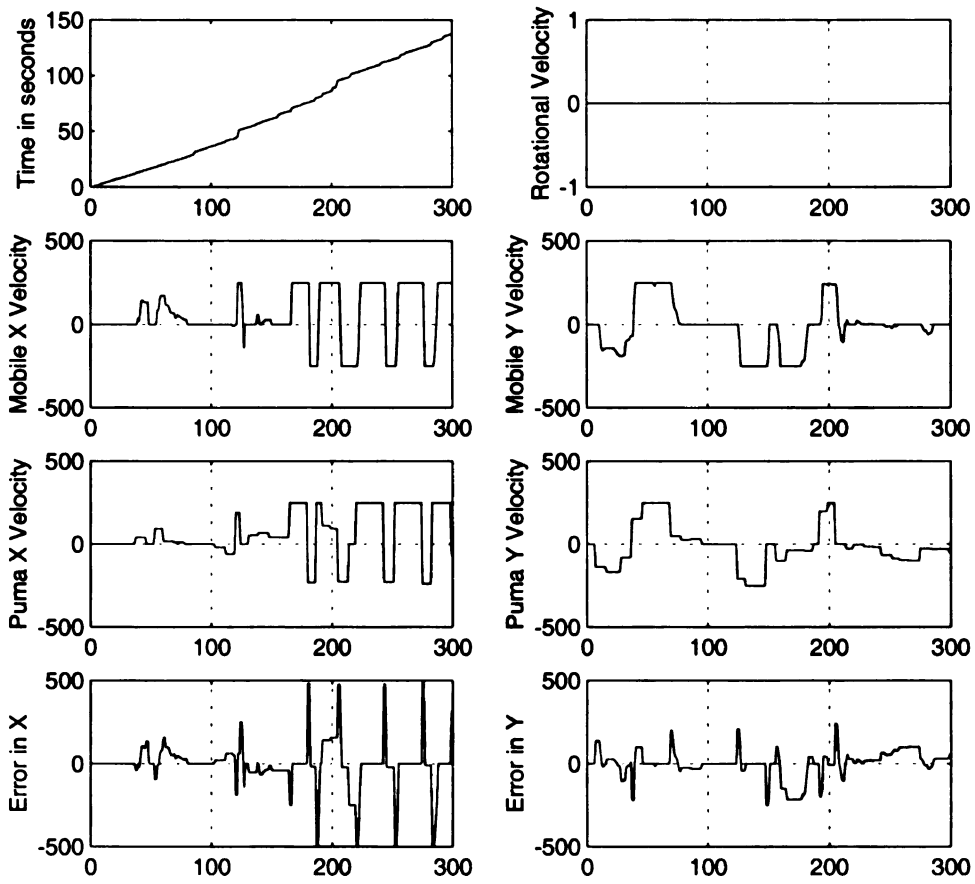

Figure 7.52: Multi-operator at multi-site collaborative teleoperation system behavior while being controlled from Hong Kong and Japan.

plot of time versus s, the event-based reference. It is clear that s is a non-decreasing function of time, which is crucial for system stability. The other plot in the top row shows the desired rotational velocity. The second row shows the desired velocities of the mobile in x and y directions, V_m. The Third row plots the desired velocities of the puma in x and y directions, V_p. The plots of V_m and V_p also correspond to the forces fed back to operator2 and operator1 respectively. The last row is the error between the mobile and the puma desired velocities in both directions. The main points to note, are the synchronization and fast response. It is clear that both robots are event-synchronized since the shift in direction occurs almost at the same s value. Fast response is clear from the sharp decrease in the error between the velocities of the two robots.

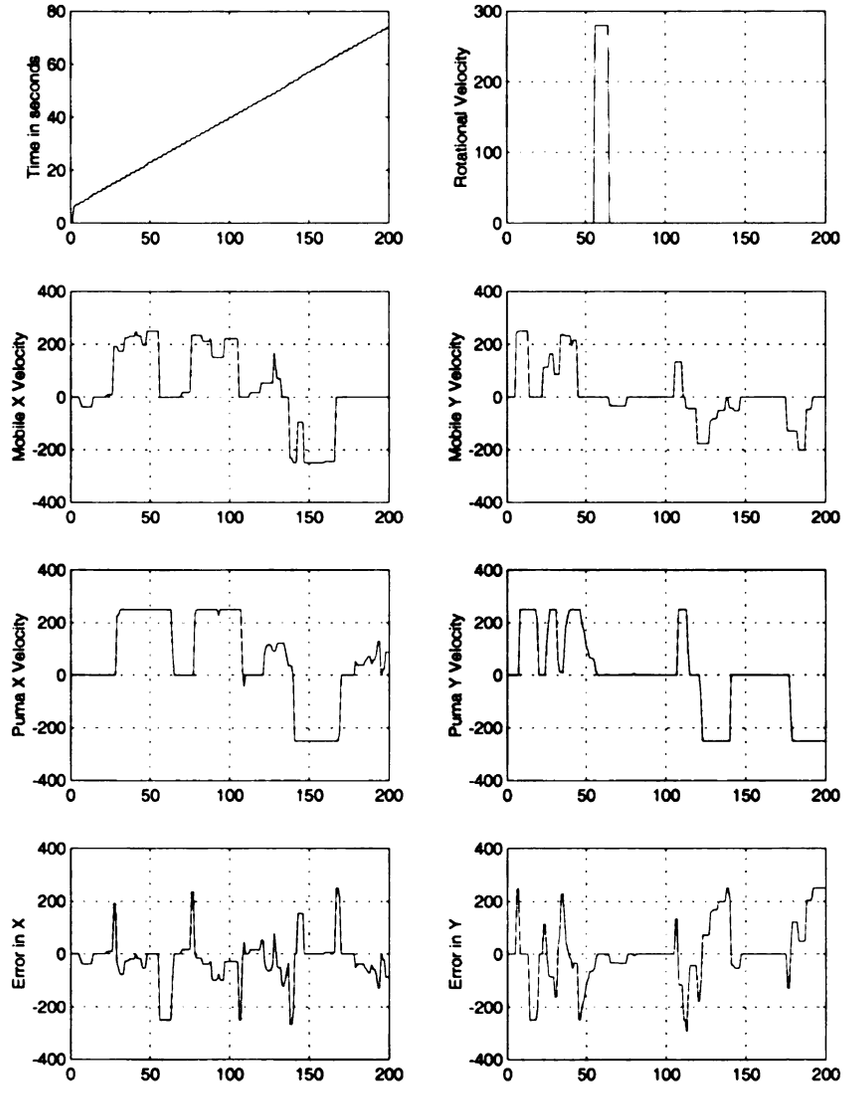

Figure 7.53: Multi-operator at multi-site collaborative teleoperation system behavior while being controlled from Hong Kong and Japan.

The results of a similar experiment are shown in Figure 7.53. In this experiment the operator in Japan is controlling the Puma (slave) and the operator in Hong Kong is operating the mobile (master). The layout of this figure and the results it shows regarding stability and event-synchronization are similar to Figure 7.52.

As for master-master collaboration, the two operators work together to achieve a certain task without having a leader. The operator is free to follow or not to based on what is decided more appropriate. Since both operators are collaborating to achieve the same job, most of the time the commands are going to be similar. The only

differences are going to occur when there exists multiple paths to the destination. In this situation one of the operators has to compromise in case a conflict is detected. A bases of this compromise might be to follow the mobile while far from the destination and to follow the puma when close to the destination. An example of this mode was experimented, where the operators' task was to navigate to a table and touch a bottle placed on it. The experiment was successfully accomplished even with low video feedback quality and despite the visual limitations.

7.8 Multi-Site Multi-Operator Tele-Coordination

This section details the implementation and experimental results of a multi-site multi-operator tele-coordination system. The experiments were conducted using two mobile manipulators as seen in Figure 7.54 [114] and according to the general architecture shown in Figure 7.55. In this setup, the operator sends position increment commands and receives force and video feedback.

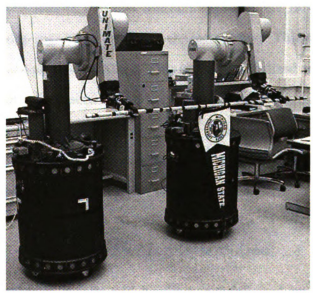

Figure 7.54: The experimental setup.

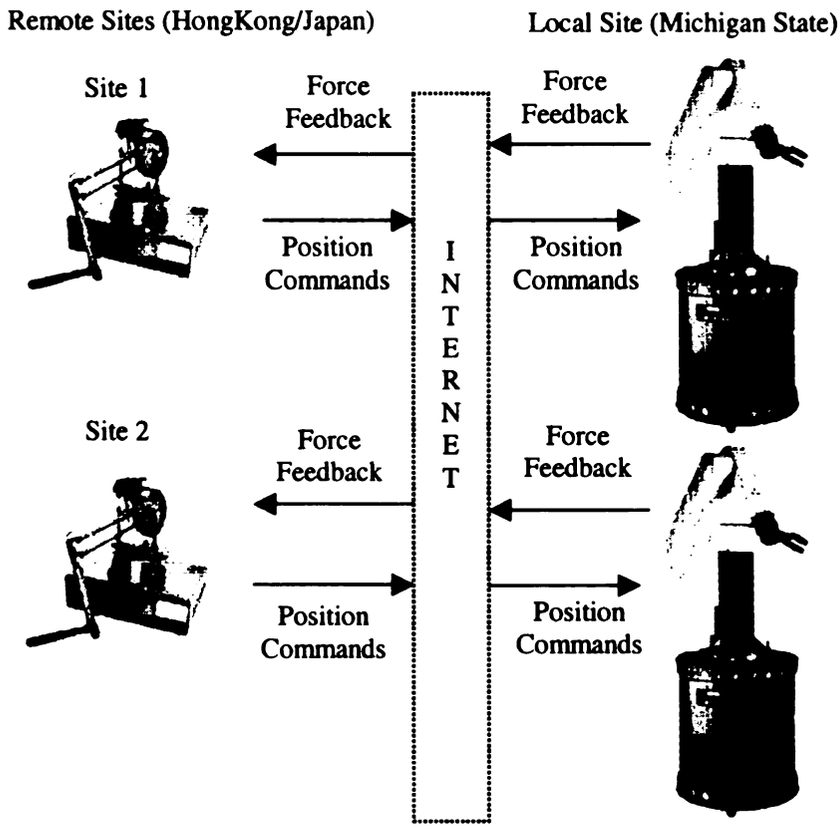

Figure 7.55: The architecture of a supermedia enhanced multi-mobile manipulator tele-coordination system.

The robots and the remote stations communicate via the Internet with no assumptions or knowledge regarding the delay characteristics. This system was tested between Michigan State and Hong Kong. This setup was utilized to create the worst scenario differential in delay faced, since the operator at Michigan State faces relatively less delay than the operator in Hong Kong. This allowed for an evaluation of the efficiency and performance of the event-synchronization algorithm implemented, which proved to increase the efficiency in the coordination.

7.8.1 System Implementation

The hardware and software details will be discussed followed by the experimental results.

Hardware

The general architecture of the system, which resembles the model shown in Figure 5.7, is shown in Figure 7.56. The two operators use Phantom haptic devices to generate 3-dimensional position increments for the mobile manipulators. Once the mobile manipulators receive the command they exchange the event reference in order to maintain synchronization. Then if the force detected is equal or greater than the maximum allowed, that is the coordination index, the direction of desired motion is compared to the direction of force. If the direction of desired motion is in the reducing force direction the command is executed else it is discarded. Meanwhile, the 3-dimensional force detected is fed back to the operator to be rendered using the Phantom device. The various hardware used has similar specifications to the ones discussed in previous sections.

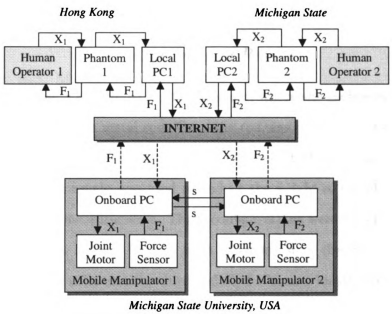

Figure 7.56: The architecture of the multi-operator multi-robot tele-coordination system implemented.

Software

Most of the software used in this experiment is similar to the one described in Section 7.6. The only exception is that the manipulator server monitors the force detected and in case it is larger than the coordination index the motion is stopped. New commands will only be executed in case they are in the force's decreasing direction. Another difference is the added synchronizing servers running on each of the robots. In this implementation, instead of the clients connecting to the PUMA server directly, they connect to a synchronizing server. This server receives the commands from the client, waits for the other server to receive the commands from the other client, and then forwards the commands to the PUMA server. The synchronizing servers execute a two-way hand shake in order to synchronize at each command arrival. In turn, the PUMA servers feedback the force to the synchronizing servers, which in turn forward this force to the corresponding clients. In addition, visual feedback was supplied from two different view angles.

7.8.2 Experimental Results

Several experiments were conducted via the Internet, where the mobile manipulators picked up and manipulated a metal rod. Once the rod was picked up, the two robots calibrated the force sensors in order to deduce the weight of the object carried by each of them. Then this weight was used to deduct the weight of the object and the inertial forces from the measurements. Three configurations were experimented, the first had the two robots holding the rod from the same side, the second had them holding it from opposite sides and the third included a human locally applying forces to the rod. During the first two scenarios the operators had a predefined pick and place task to be accomplished and in the third scenario the operators were simply required to follow the local human applied forces.

Figure 7.57 shows the results using the second scenario with the robots on oppo-

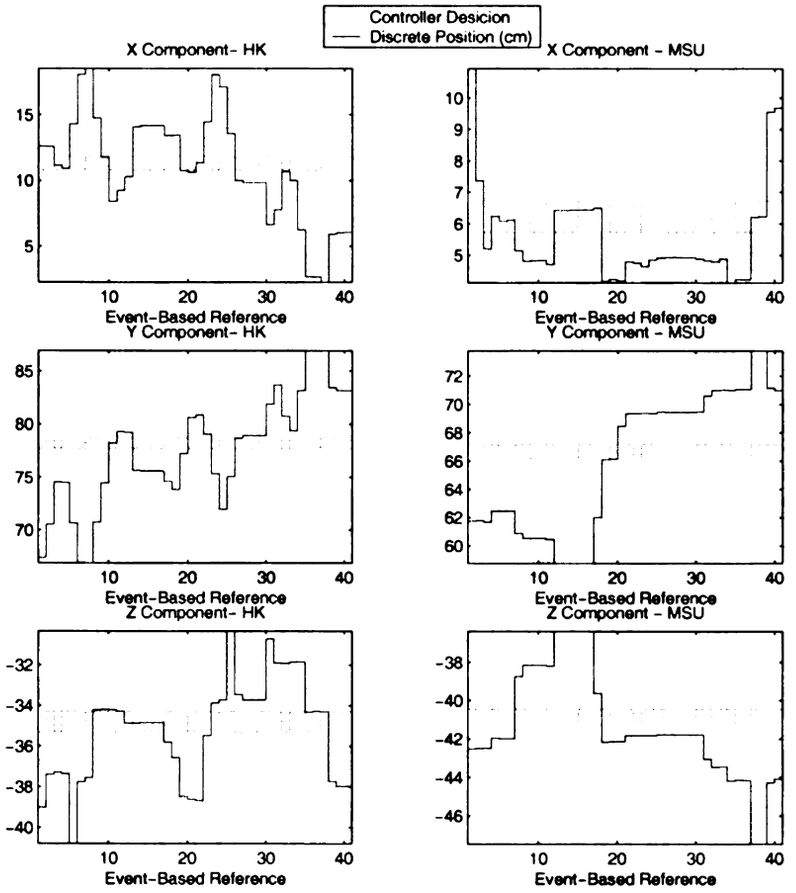

Figure 7.57: Tele-coordination results of two mobile manipulators moving a metal rod.

site sides of the rod. The first column of figures depicts the data from the mobile manipulator controlled from Hong Kong and the second column depicts these from the one controlled from Michigan State University. The x, y and z components are plotted with respect to the event-based reference in rows 1, 2 and 3 respectively. The gray line in these plots indicates whether the controller allows motion at that event-based reference. A high value of the waveform corresponds to the case where the force detected is less than the coordination index or the desired motion is in the direction of force reduction and thus motion is permitted. A low value of the waveform corresponds to the case where the force detected is equal or larger than the coordination index and the desired motion is in the direction of increasing force and thus motion is not permitted. The dark line in these plots shows the distance traveled at that

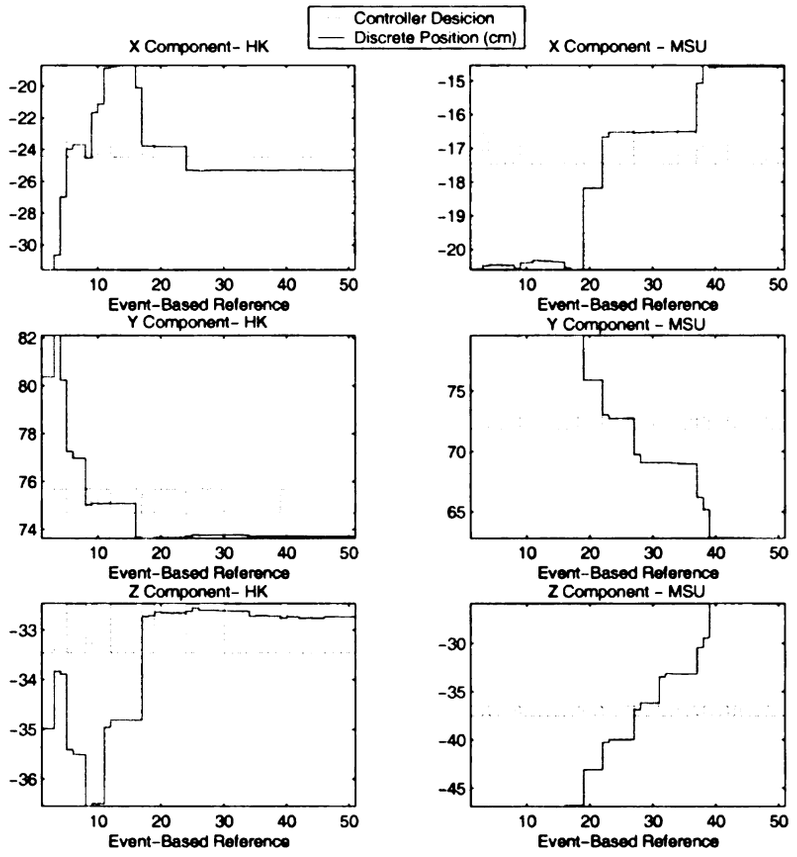

Figure 7.58: Tele-coordination results of two mobile manipulators moving a metal rod while a human is applying direct forces to the rod.

particular event.

The task was accomplished successfully and as expected, there is no motion at events where the controller decision is low corresponding to no permitted motion. In other words, changes in position occur only during the controller high state. The only discrepancies are due to measurement errors and noise and they do not exceed a few millimeters. So tele-coordination was accomplished with a prespecified coordination index.

Figure 7.58 shows the results using the third scenario with the robots on the same side of the rod and a human applying forces to the rod. The layout and results are similar to the ones presented in Figure 7.57.

Figure 7.59 shows the results from an experiment done according to the third scenario, where the operators were required to follow the force applied by a local

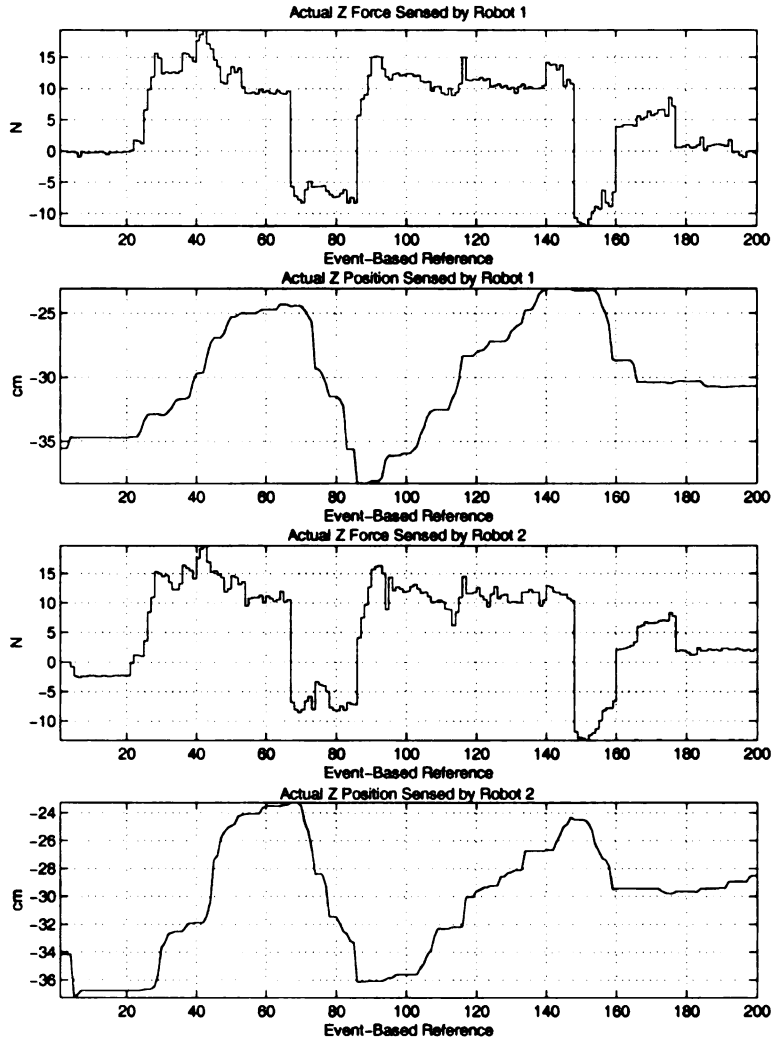

Figure 7.59: Two mobile manipulators position tracking to the force applied by a human on the manipulated object.

human on the object being manipulated. Plots in rows 1 and 3 are the forces in newton detected by each mobile manipulator in the z axis with respect to the event-based reference. The plots in rows 2 and 4 are the corresponding z positions of the robots in *cm* with respect to the event-based reference.

As seen for both robots the z positions shown in rows 2 and 4 are tracking the forces sensed shown in rows 1 and 3. The longer the force is applied in a certain direction the more significant the motion is for both robots. Therefore, the two mobile manipulators are capable of tracking external forces while maintaining the

required coordination index.

The experimental results show that under the control strategy proposed multi-robots can successfully tele-coordinate while maintaining a certain coordination index irrespective of the delay faced. This is also independent of the configuration used, the external forces applied or the object being manipulated.

7.9 Tele-autonomous Control of Robot Formations

Despite all the advances in artificial intelligence it is still inferior to human intelligence. However, there are situations in which the artificial intelligence excels and can be used to free humans from tedious tasks. Therefore, a combination of both human and machine control is ideal for many scenarios [115]. In addition, the efficiency of operation can be significantly increased by employing multiple robots in a formation and by enhancing the operator's coupling with the environment by supplying supermedia feedback [116]. A general setup of such a system can be seen in Figure 7.60.

Several applications from different fields motivate this research [116]. Military applications are some of the driving forces for this technology. One operator capable of remotely monitoring and controlling, if needed, several vehicles is a significant part of future combat systems. Another application is search and recovery type of tasks, where one operator is capable of scanning a wider area by deploying multiple robots in a formation and monitoring the sensory feedback from all of them. Once a "hot spot" is sensed the formation can be converged towards the target using teleoperation. Industrial applications also exist; for instance, inspections of large facilities can be done using formations of robots assisted with human intervention if required. In addition, clean up and surveying type of tasks can be made easier and more efficient using such technology.

Combining advantages from different technologies also combines the challenges

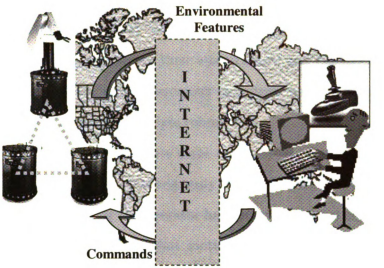

Figure 7.60: The general architecture of a tele-autonomous formation control system.

of these technologies. The challenges faced are these of formation control and teleoperation, specifically over the Internet. The challenge of formation control is the establishment of a coordination scheme for heterogeneous robotic systems. The difficulty faced in accomplishing this relates to the uncertainties of the environment.

In addition, there is the challenge of control transition from autonomous operation to teleoperation and vice versa, which has to be done on the fly with no reinitialization or replanning required. In most existing planning and control schemes for tele-autonomous operation, autonomy and teleoperation are mutually exclusive modes of operation. That is the system is either in autonomous mode or teleoperation mode and once it is in teleoperation mode it requires complete replanning to switch back to autonomous operation. To overcome the difficulties, event-based planning and control is used for the formation and the teleoperation [117].

Tele-autonomous operation, which is considered in this research, combines autonomous operation and teleoperation. This approach takes advantage of the speed, repetitiveness and accuracy of autonomous operation and the flexibility, creativity and ingenuity of the human intelligence. This approach is also known as watch-and-intervene since the operator's main job is to monitor the formation and take action

only in cases where the system faces unexpected situations. This reduces the work load of the operator and thus increases their attentiveness and efficiency. Combined with formation control, under tele-autonomous operation, one person is able to simultaneously monitor and control multiple robots.

In a tele-autonomous system there are three main modes of operation; either the formation is operating completely autonomously, or the formation is sharing control in real-time with the operator, or the operator has complete control of the operation.

Let the autonomous function of robot motion be described by $Y^d(s)$ and the human generated commands from the joystick be described by $Y_h^d(s)$, where s is the motion reference parameter. Based on the three modes of operation discussed previously the resulting command, which is a combination of the autonomous and the supervisory commands, can be expressed as such:

$$Y_{res}^d(s) = \begin{cases} Y^d(s) \\ Y^d(s) + Y_h^d(s) \\ Y_h^d(s) \end{cases} \tag{7.1}$$

The first case is pure autonomy and the desired command is based on the autonomous plan. In this case no operator intervention is required and thus $Y_h^d(s) = 0$. However, once an unexpected event in encountered the system might request the supervisor's help or the supervisors themselves might elect to intervene. This is decided based on the state of the formation and the feedback sensed by the operator. In the case of an intervention there are two possibilities; either the command is a combination of the autonomous plan and the operator's commands or it is entirely based on the supervisory commands.

7.9.1 System Implementation

The formation control was implemented on a mobile robot and mobile manipulator setup, which included teleoperation capabilities. During tele-autonomous operation, the operator sends velocity commands to the formation and a force is fed back. This force is given by

$$F = K/d \tag{7.2}$$

where d is the shortest distance to obstacles in the direction of motion and k is a scaling factor. It is clear that the force in this case is inversely proportional to the distance to obstacles. So when the robot formation gets closer to an object the force sensed increases accordingly.

Hardware

The hardware architecture of the formation tele-autonomous operation system is shown in Figure 7.61. The system is composed of a mobile manipulator and a mobile robot. The joystick used to control the formation is an MS force feedback joystick. The specification of the different hardware components is similar to what have been discussed in the previous sections.

Software

The software developed for this system consists of three main routines: formation client, mobile/manipulator server and mobile server.

Formation Client: In case human intervention is required, this client sends velocity commands to the formation via the mobile/manipulator server. It is also responsible for receiving force information from the mobile/manipulator server and

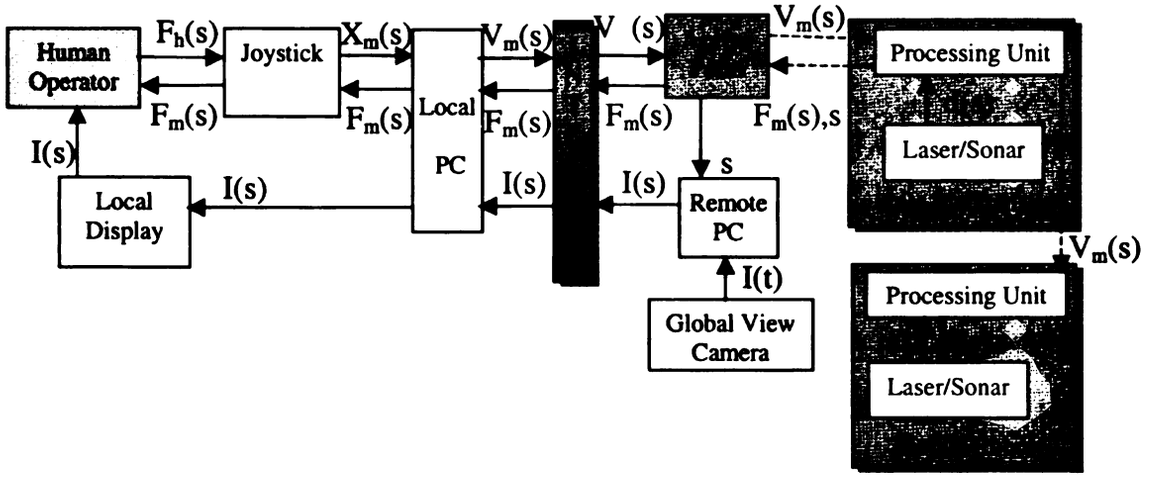

Figure 7.61: The architecture of the tele-autonomous robot formation system developed.

rendering it.

Mobile/Manipulator Server: This server runs on the mobile/manipulator, which is the front robot in the formation. In case there is no human intervention, this server executes autonomously the original desired path. If the human operator intervenes, this server receives velocity commands from the client, forwards these commands to the mobile server, controls the mobile manipulator and feeds back the force information to the client. Control is done according to the general obstacle avoidance algorithm.

Mobile Server: This server runs on the mobile robot, which is the one at the back of the formation. It receives the velocity commands from the mobile/manipulator server and controls the mobile robot. The commands are either autonomously generated or based on the human remote control. Commands are modified in order to maintain a fixed distance between the two robots and to avoid obstacles. In case this robot is blocked a procedure exists for it to inform the front robot to wait [117].

As with all the experiments, VIC is used to feed back several video streams from the environment.

7.9.2 Experimental Results

The result of tele-autonomous operation is shown in Figure 7.62, the robots move autonomously in a sine wave formation until an obstacle is detected. At this point the operator intervenes to move the formation away from the obstacle.

Plots (a) and (b) of the figure show the trajectories of both robots, plots (c) and (d) show the closest distance detected by the front robot with respect to time and plots (e) and (f) show the forces fed back to the operator in the x and y directions respectively both with respect to time.

These plots show that once an obstacle is detected, that is the distance is closer than a certain predefined value, as shown in plots (c) and (d), the force increases in the direction opposite to that of motion as shown in plots (e) and (f), where both forces increase as an obstacle is detected. In this case, the operator takes control on the fly and teleoperates the formation away from the obstacle. Once teleoperation is stopped the formation autonomous motion is resumed as seen in plots (a) and (b). These results confirm the advantages of the perceptive planning and control scheme, where autonomous formation control can be established with minimal error, where the formation can handle unexpected events with no need for replanning or resynchronization and where tele-autonomous operation can be established on the fly with no need for control hand over or replanning.

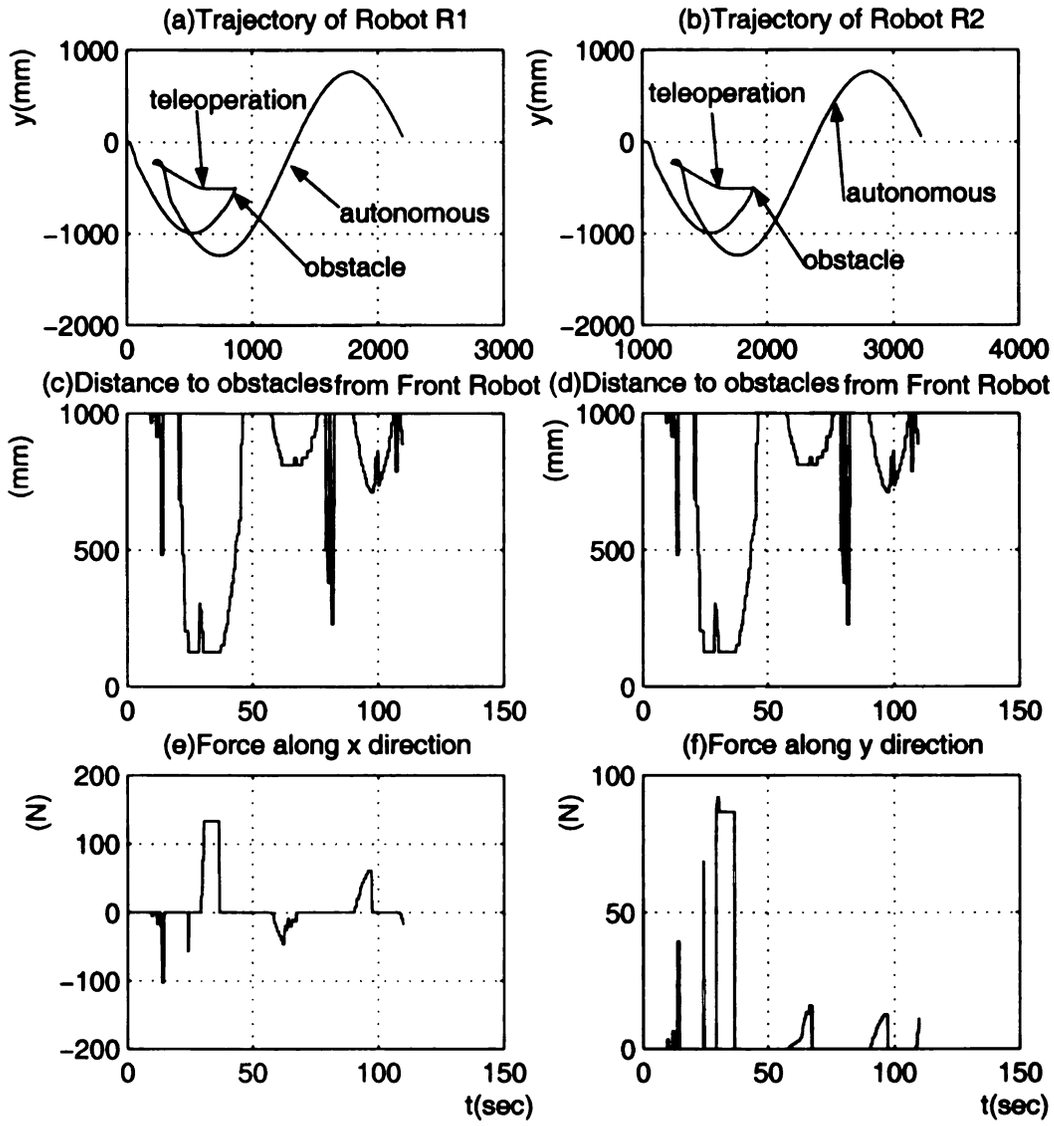

Figure 7.62: Experimental results under tele-autonomous operation with obstacles

7.10 Supermedia Enhanced Teleoperation

Some applications require other types of feedback than force and video. These might required supermedia to be fed back; such as, temperature and pressure. Since the theory developed in this research does not depend on the specific system, then event-based control and planning can be used to design systems that satisfy these requirements. To illustrate the generality of the theory developed, a supermedia enhanced

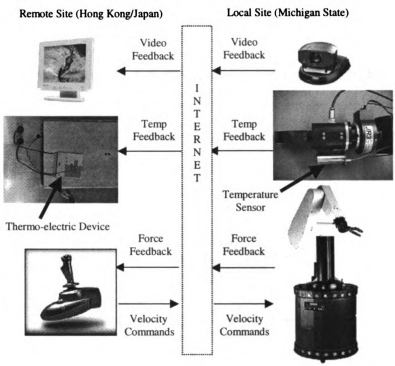

Figure 7.63: The architecture of a supermedia enhanced teleoperation system.

teleoperation system was developed similar to the one shown in Figure 7.63.

The operator sends velocity commands, which the mobile manipulator executes. This velocity is divided by the local controller between the arm and the mobile base. This division is done to achieve an acceptable posture for the arm as discussed before [114]. The haptic feedback in this case is actual force detected by the force/torque sensor mounted on the gripper. Also temperature and real-time video are fed back. All the supermedia are rendered to the operator in their original forms.

7.10.1 System Implementation

The hardware and software developed for this system are presented in this section.

Hardware

The experiments were done using a mobile manipulator according to the system architecture shown in Figure 7.64. The hardware required for the mobile manipulator teleoperation is similar to the one discussed previously. This implementation also included the remote temperature sensing and rendering system.

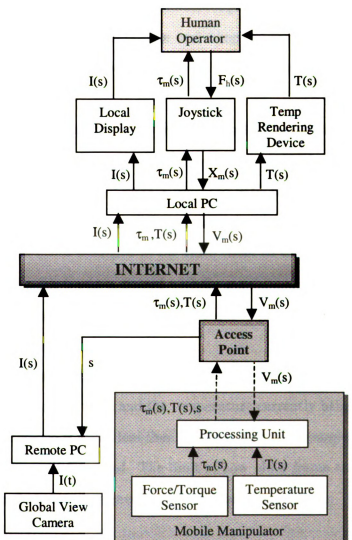

Figure 7.64: The architecture of the supermedia enhanced teleoperation system developed.

Software

The programs required for this system are the mobile/manipulator and temperature server, mobile/manipulator and temperature client, video server and video client.

Mobile/Manipulator and Temperature Server: This server runs on the robot and is responsible for receiving the velocity commands and deciding the appropriate control for the manipulator and the mobile base. It is also responsible for sampling the temperature sensor and the force/torque sensor in order to feedback this information to the mobile/manipulator and temperature client. In addition, it forwards the event number to the video server to be used on the client side for synchronization.

Mobile/Manipulator and Temperature Client: This client runs on the remote machine. It forwards velocity commands to the server and receives force and temperature information back. The force is rendered with the joystick and the temperature is sent to the temperature rendering device connected to the serial port. Also this client forwards the event reference to the video client so that it can be compared with the reference that the frame is tagged with for synchronization purposes.

Video Server: This server tags each video frame with the event reference and forwards it to the video client.

Video Client: Before this client displays a frame it compares its event reference with the event reference of the force and the temperature currently being rendered. If the frame's event reference is much less than the force's and the temperature's event reference then the frame is discarded. The limit of how old a frame can be and still be displayed is a performance parameter which has to be tuned.

7.10.2 Experimental Results

The performance of the system was experimented under no delay and random delay conditions. The mobile manipulator was teleoperated locally from Michigan State

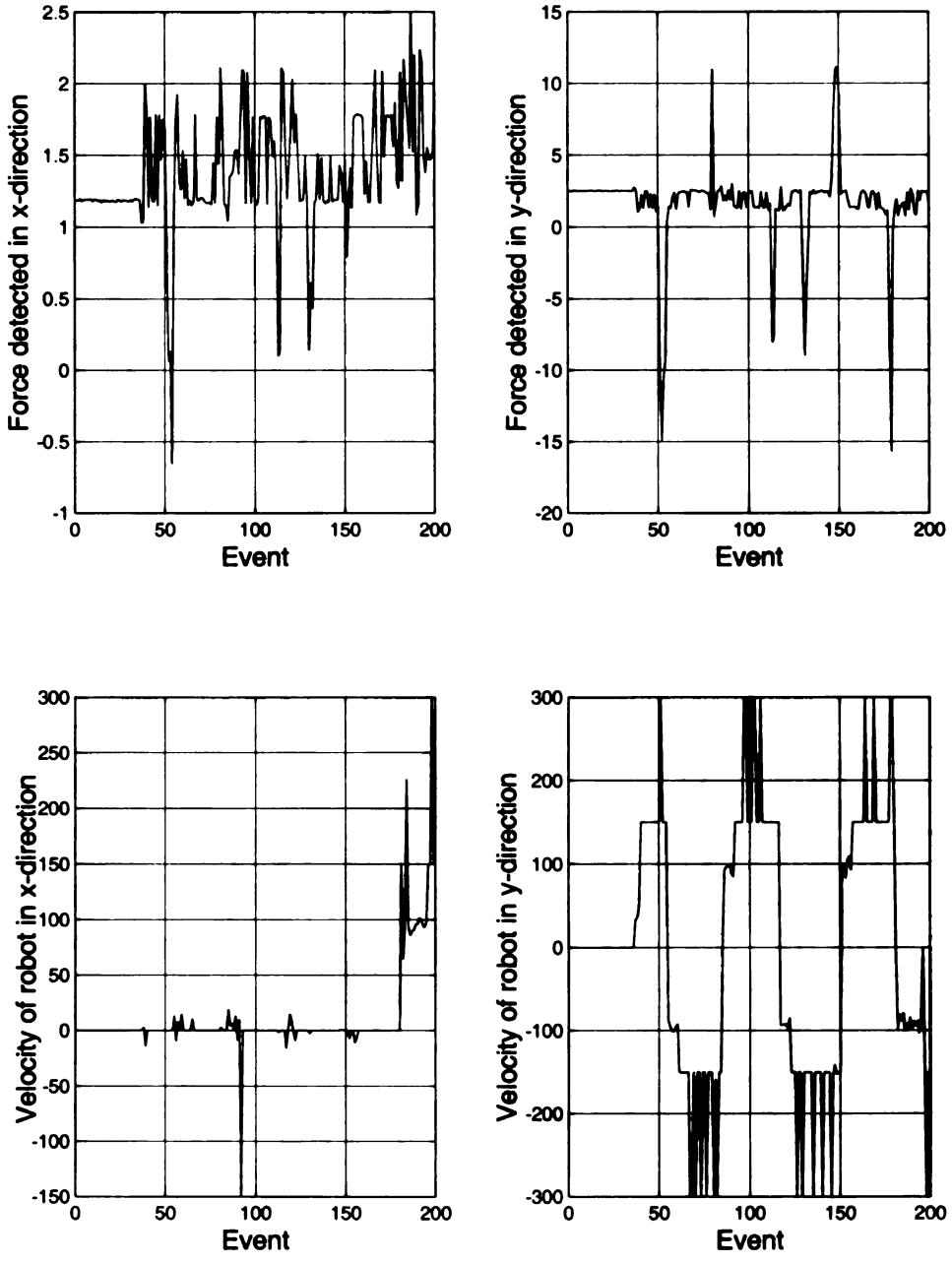

Figure 7.65: Performance of the system while being controlled from Michigan with no considerable delay.

University and remotely from the Chinese University of Hong Kong. The operators were asked to move the robot is any direction and once they detect a force they were asked to command the robot in that direction. The results are shown in Figure 7.65 for the no delay case and in Figure 7.66 for the random delay case. The first row in each figure shows the forces detected in the x and y directions with respect to the

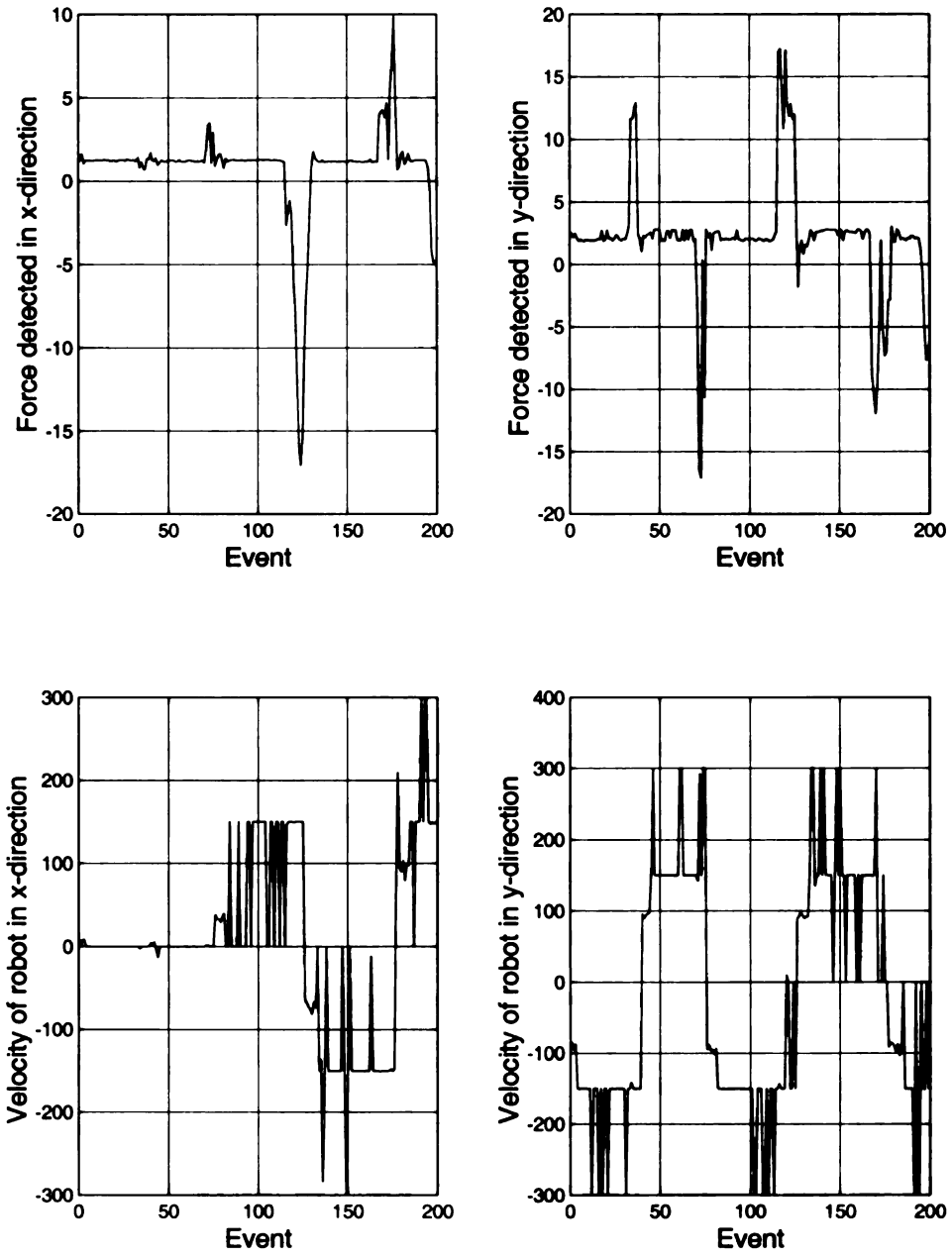

Figure 7.66: Performance of the system while being controlled from Hong Kong with random delay.

event reference, the second row gives the actual velocity of the robot in the x and y directions with respect to the event reference. The event-based action reference in this system is taken to be the distance the robot moved.

As seen, the velocity of the robot in both cases is responding to the force similarly. Once a force is applied on the robot as seen in the first rows the velocity

changes direction to match that force as seen in the second rows. For example when a force is applied in the y direction the velocity changes to the y direction. Although different operators did these experiments, the performance is similar in both cases. This illustrates the implications of event-transparency, which states that if a system is event-transparent then the control is consistent regardless of random delay. In addition, it is clear in both figures that the control signal, which in this case is the velocity, is a reflection of the most current state of the system, that is the force detected. This is seen since there is no difference between the reference at which the force is detected and the reference at which the velocity responds to it. This implies that the system satisfies part of the event-synchronization requirements.

As for the event-synchronization requirements that relate to the supermedia being rendered, the results are shown in Figure 7.67. This figure gives the temperature sensed on the robot, received remotely and rendered to the operator with respect to the event reference in the first column. It also gives the forces detected and received with respect to the event reference in the second column. As seen the received signals are identical to the detected ones with respect to the event reference. Also all the supermedia streams are event-synchronized, which implies that the force and temperature detected at a certain event reference value are rendered at the same event reference value. For example, the force and temperature detected at event reference s_n are rendered to the operator at the same event reference s_n. Note that the rendered temperature plotted is slow with respect to the detected temperature. This is due to the fact that the external temperature sensor used to detect the temperature of the rendering device has slow response.

Similarly, the video is event-synchronized with the force and the temperature. However, since video feedback is slower than force and temperature feedback, a certain tolerance has to be allowed. So the frame generated at s_n can be rendered at any event reference within $s_n + N$, where N is the accepted tolerance. If the frame does

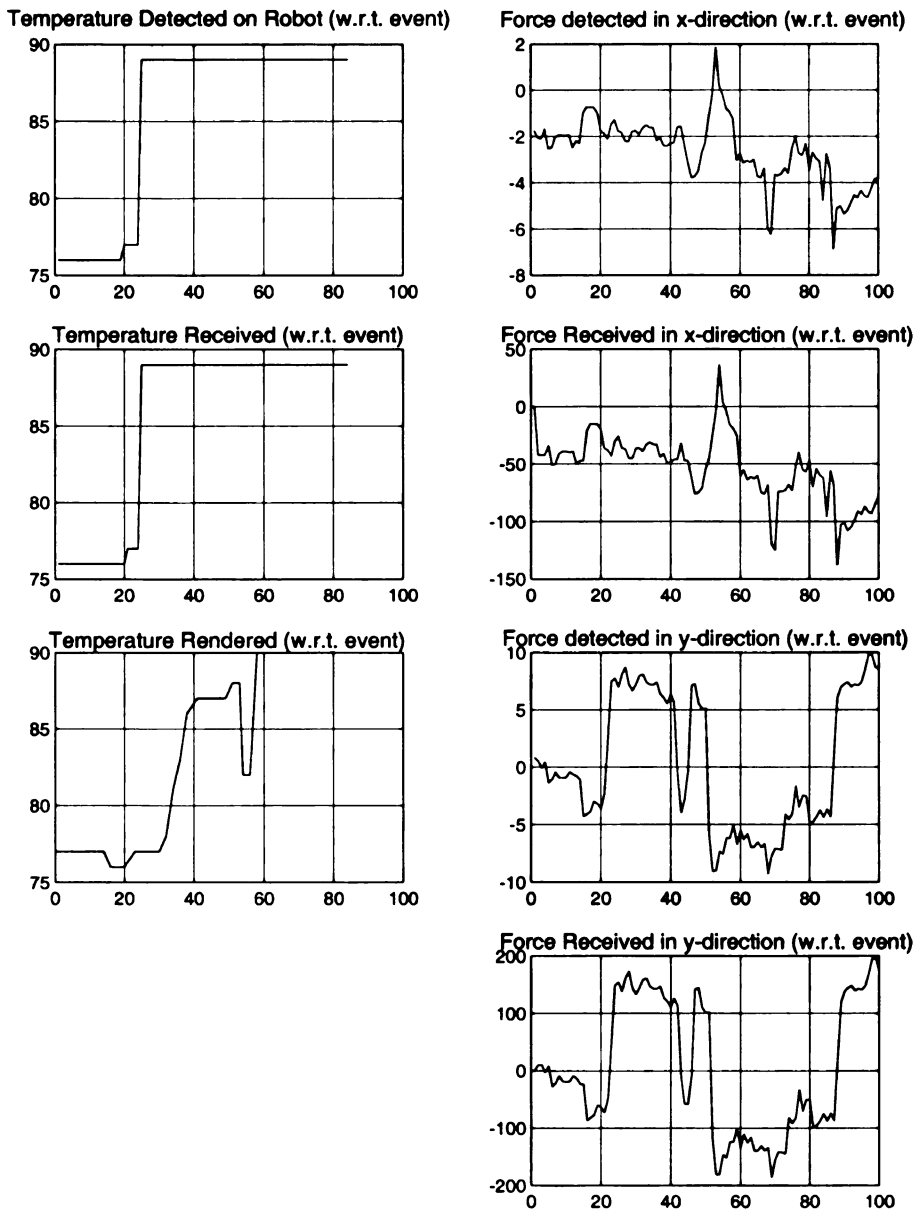

Figure 7.67: Performance of the system while being controlled from Hong Kong with random delay.

not arrive within this margin then it is discarded. So what the operator will be seeing cannot be older than N events from what is being felt. This is seen in Figure 7.68, where the first row shows the force and the frame numbers with respect to the events at which they were sampled and the second row shows the force and the frame numbers with respect to the events at which they were rendered. It is shown that the frames were either rendered within 10 events of the event at which they were sampled

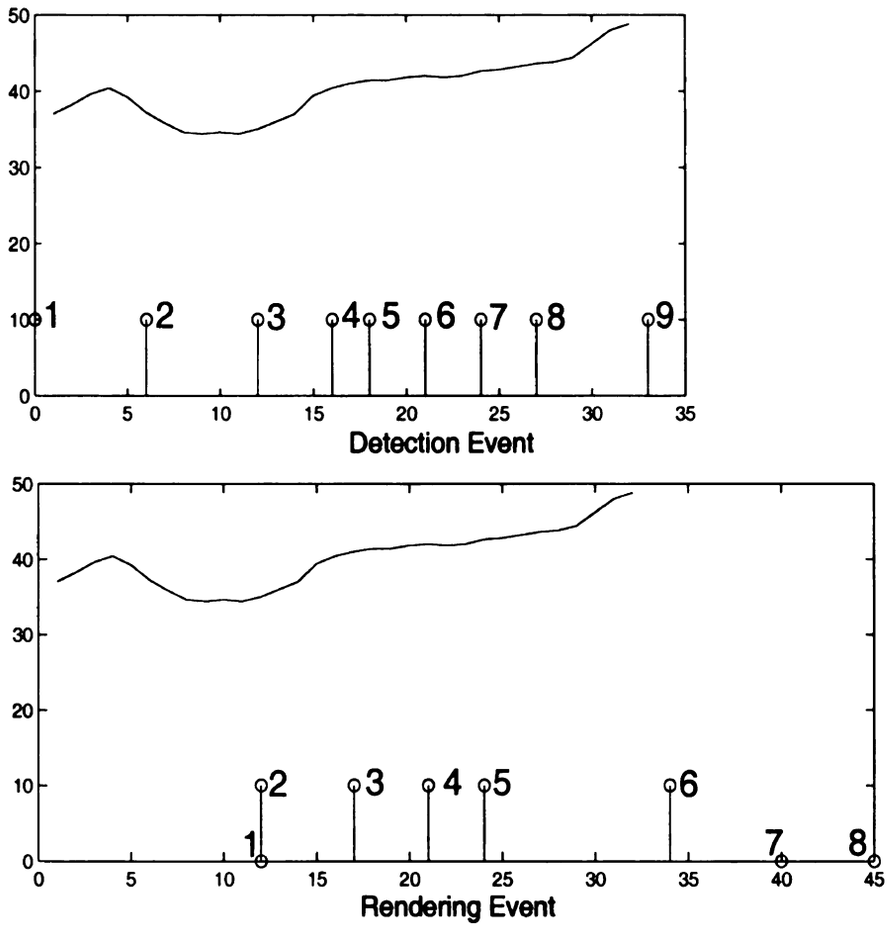

Figure 7.68: The sampling and rendering of the video frames and the force with respect to the event-based action reference.

and that is shown by a bar, or they were discarded and that is depicted by a circle on the axis, such as the ones for frames 1, 7 and 8. This results in the video being event-synchronized with the other supermedia streams with a certain tolerance.

The experimental results presented confirmed the analysis made regarding event-transparent event-based control teleoperation systems. This implies that performance of the system is consistent in the face of random time delay. Also the supermedia streams fed back and the control signal are event-synchronized.

7.11 Micro Manipulator

Micro Mechanical and Electronic Systems (MEMS) technology makes it possible to sense and act in micro-environments [118] [119]. Combining the Internet and MEMS it is possible for humans to sense and act in a remote micro-environment. This new technology has potential impact on several fields; such as, biomedical engineering and manufacturing. However, in order to avoid breaking or damaging micro objects during the manipulation process, force reflection is an essential component within the control structure of micro-manipulation and micro-assembly systems.

The supermedia feedback (force and video) supplied in this experiment, link and couple the operator to the environment. This coupling increases the efficiency and safety of manipulation at microscopic levels. It is similar to magnifying not only the visual but also the sensing capabilities of the operator. Therefore, the force, which conveys significant information to the operator with minimal overhead, is a desirable form of supermedia feedback in micro-teleoperation systems.

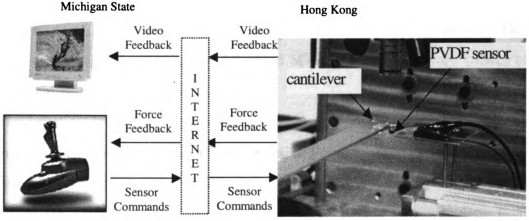

Figure 7.69: The structure of a micro-manipulator teleoperation system.

A micro manipulator teleoperation system with force feedback is shown in Figure 7.69 [120]-[122]. This section gives the implementation details and experimental results of such a system.

7.11.1 System Implementation

The implementation discussed in this section concentrates on the teleoperation part of the system and does not include much details about the micro manipulator or micro sensor [120]-[122].

Hardware

The hardware architecture of the system is shown in Figure 7.70. The joystick is the MS force feedback similar to the one used in the previous experiments. The hardware includes a camera capable of feeding video from a micro scale. In addition, a PVDF (Polyvinylidence fluoride) micro-tip was used as a piezoelectric sensor for the force [120]-[122]. This tip is about 2.5 millimeter long with about 0.8 millimeter at the triangular base. The output from this sensor is amplified using an inverted amplifier with feedback gain of 50. Its signal is then fed to a 8255 analog-to-digital conversion (ADC) card connected to a PC for signal transmission to the Internet. This experimental setup is located at the Advanced Microsystems Laboratory (AML) of The Chinese University of Hong Kong.

The sensor tip is attached to an x-y computer-control positioning table, which can be controlled via the Internet by a force reflection joystick in the Robotics and Automation Laboratory (RAL) at Michigan State University. A cantilever is attached to a vibration drum and has a tip vibration of 100 millimeter to 1 millimeter from the frequency range of $1Hz$ to $120Hz$. The AML sensor tip position can be manipulated by the RAL joystick to contact the vibrating cantilever. The RAL operator observes the AML tip position using a video conferencing software. The force of the

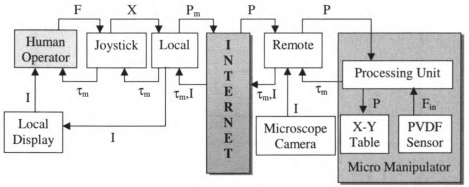

Figure 7.70: Hardware architecture of the micro-manipulator teleoperation system.

vibrating cantilever sensed by the tip is sent to RAL via the Internet. Once the force is received the force feedback joystick renders it. After that the operator generates a new movement command to be sent to the sensor via the Internet.

Software

The software developed is a motion sever and a client.

Motion Server: This server receives the position increment commands from the client in both x and y directions. These are forwarded to the x-y table for execution and at the same time the sensor is sampled for the force in the y direction. The force is then sent to the client.

Client: The client translates the joystick position into increment commands and sends these to the server. It also receives the force feedback and renders it to the operator.

In addition, real-time video was fed back from the micro-environment using VIC.

7.11.2 Experimental Results

The experimental results presented here relate to the testing done between Hong Kong and Michigan State. Figure 7.71 shows a plot of the desired position increments in both directions, in the first two rows, and a plot of the rendered force with respect

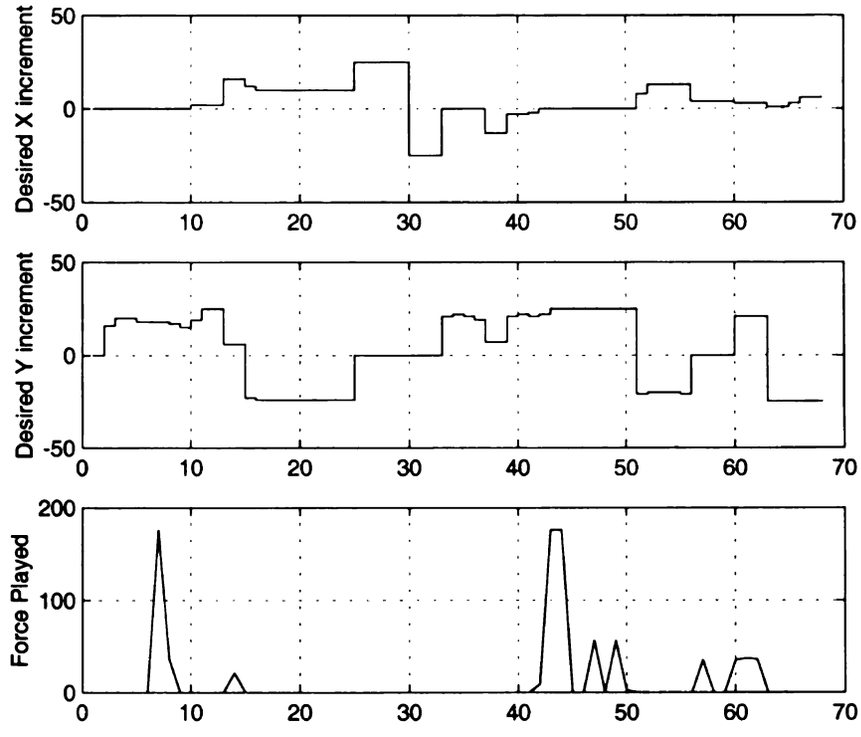

Figure 7.71: Plots of the desired position increments and the force felt by the operator.

to the event, in the third row.

The commands sent are random, which is typical of a teleoperation scenario. This makes approaches based on prediction of forces or virtual forces non-realistic. Therefore, actual force had to be sensed and fed back. Figure 7.72 presents plots of the force felt by the operator in the first row, the force sampled from the sensor in the second row and the error between them in the third row all with respect to the event-based reference. As seen, the force felt is closely following the one sampled from the sensor, which implies that the system is stable. Also the force rendered responds at a close event to the changes in the force sensed, which reflects event-synchronization.

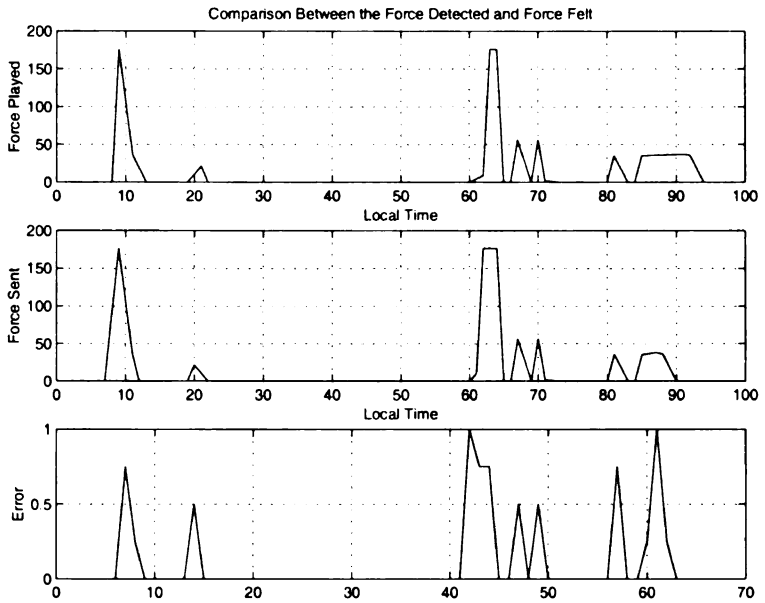

Figure 7.72: Comparison between the force felt and the one sensed.

7.12 Summary of Experimental Results

In summary, all these experiment show that the event-based planning and control developed as part of this research is a general framework for the design of real-time supermedia enhanced teleoperation systems. This framework is independent of the specific robot used, the human operator, the environment model or the communication medium.

In addition, the experimental results reveal that event-based controlled real-time teleoperation systems maintain stability in the face of random delay. Also these systems are event-transparent, which offers consistency in the performance, and event-synchronized, which increases efficiency and safety. So this framework ensures synchronization between the operator and the robot and between the different supermedia streams being rendered.

The next chapter summarizes this dissertation, gives concluding remarks and details the future work.

CHAPTER 8

SUMMARY, CONCLUSIONS AND FUTURE WORK

This chapter gives a summary of this document. It also gives concluding remarks and observations regarding the research conducted. Then additional improvements to the approach and future work will be suggested.

8.1 Summary

This dissertation started by motivating Internet-based teleoperation and specifically these that include supermedia feedback. It also discussed the many challenges and difficulties faced in designing and implementing such systems. These challenges that include random delay, human-machine interfacing, non-determinism, data loss and others. Also a detailed literature review was given that covered most aspects of this research.

Then the Internet as a real-time control media was examined. The characteristics of Internet communication and their effects on teleoperation systems performance were covered. Effects such as instability, loss of transparency and desynchronization.

This was followed by the contributions of the research done, which can be divided into theoretical and experimental. The theoretical contributions included the following:

- Event-based planning and control for real-time bilateral teleoperation systems was introduced. This control method was analyzed in terms of stability. Also event-transparency and event-synchronization concepts were developed and their implications analyzed. It was shown that event-based teleoperation systems, under certain conditions, satisfy stability, event-transparency and event synchronization regardless of the randomness of delay.

- The modeling of event-based bilateral teleoperation systems using dynamic equations was detailed. Also, Petri Nets were proposed as a modeling and analysis tool for such systems. Then these modeling and design approaches were compared with another popular method in the literature.

- The design methodologies that guarantee stability, event-transparency and event-synchronization for bilateral teleoperation systems were included. A new design approach based on Petri Nets was developed for event-synchronization.

- The importance of the developed concepts were illustrated by studying bilateral tele-coordination of multi-robots. The concept of coordination index for bilateral teleoperation systems was developed. The design methodology for establishing tele-coordination with a certain small index was detailed.

- All the theory developed is independent of the time delay faced and applies to a wide range of systems operating in unknown environments with different human operators, tasks and types of feedback. All the methods do not require significant overhead and are easy to implement.

As for the experimental contributions they included the following:

- Several different event-based bilateral teleoperation systems were implemented. To the best of our knowledge these were one of a kind systems that have not been developed previously.

- Several types of robots ranging from mobile manipulators to micro manipulators were bilaterally controlled with different types of supermedia feedback, which included haptic, temperature and video. To the best of our knowledge this was the first time haptic and temperature information were fed back via the Internet.

- Experiments were carried out via the Internet using test beds that included The United States, Hong Kong and Japan.

- The experimental results confirmed all the theory developed and illustrated its benefits in the face of random nondeterministic delay.

8.2 Conclusions

Teleoperation is a very attractive and quickly growing research field. But this technology faces many difficulties. The most important one, which will always exist, is time delay. At first thought this might seem as a trivial issue not affecting the system performance. But it was shown that this delay would render the system unstable and degrades the system performance, especially when supermedia (sensory information: video, audio, haptic, temperature and others) is fed back.

The work found in the literature has several limitations when it comes to supermedia enhanced teleoperation under nondeterministic delay conditions. Those limitations mainly relate to assumptions made regarding the delay; some take the delay to be constant others do not even consider delay.

This research extended event-based control in order to apply for supermedia enhanced real-time teleoperation systems. It was shown theoretically and experimentally that this approach results in a stable teleoperation system regardless of the time delay encountered. In addition, the notions of event-transparency and event-synchronization were developed. The theory and experimentation revealed the implications of these performance properties. Implications which include consistency and efficiency in the control despite random delay.

Moreover, modeling for such systems was accomplished using Petri Nets, which were also used for analysis and design. Detailed design procedures for stability, event-transparency and event-synchronization were given. The importance of those char-

acteristics is illustrated using the concept of tele-coordination. It was shown that tele-coordination under certain performance conditions can be achieved by event-transparent and event-synchronous systems.

The major contribution of this research is the generality of the theory developed. The approach developed and the analysis done are independent of the specific system being controlled. Also they are independent of the environment model, the delay model and the human operator. In addition, this approach does not require any over-head and is easy to implement.

Therefore, as shown by the several different experimental systems developed, any type of application can adopt this approach. Micro or macro scale applications can use event-based control for supermedia enhanced real-time teleoperation via the Internet or any communication medium. To the best of our knowledge, these are the first such supermedia enhanced teleoperation systems to be controlled in real-time via the Internet.

This work will have an impact on the fields of robotics, communication, control and multimedia. It gives new modeling, analysis and design approaches that can be adapted for any of these fields. In addition, new measures such as event-transparency and event-synchronization compose an innovative way for evaluating performance. As for teleoperation specifically, this research elevates the level of hope for the feasibility of Internet-based real-time bilateral control systems. Since the research carried out is technology independent and since most of the issues investigated, such as delay, will not vanish in the foreseen future, this research will not become obsolete in the face of technological advances.

8.3 Future Work

The future work suggested relates to some topics researched as part of this project and also to some topics that were not investigated. The following are the main suggested topics:

- Additional work should be done relating to transparency from the operator's perspective. Examining how realistic the supermedia feels compared to direct interaction with the environment. This measure should be objective and easy to evaluate.

- The quality of service should be studied. Specifically, developing measures for quality of service in real-time teleoperation systems and investigating the kind of quality of service that can be guaranteed.

- This field has much potential for the use of Neural Networks. Many parameters can be adapted using Neural Networks to better match the specifics of the system. A changing event-based reference can be used to adapt to a changing system or environment.

- The use of wireless application protocol WAP should also be investigated in the remote sensing and control context [123]-[125]. Since wireless communication is a field growing at a fast pace, there is great interest in robotics applications based on WAP.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] W. Kim, B. Hannaford, and A. Bejczy, "Force-Reflection and Shared Compliant Control in Operating Telemanipulators with Time Delay," IEEE Transactions on Robotics and Automation, Vol. 8, April 1992.

[2] M. Otsuka, N. Matsumoto, T. Idogaki, K Kosuge, T. Itoh, "Bilateral Telemanipulator System With Communication Time Delay Based on Force-Sum-Driven Virtual Internal Models," IEEE International Conference on Robotics and Automation, pp. 344-350, 1995.

[3] G. Leung, B. Francis, J. Apkarian, "Bilateral Controller for Teleoperators With Time Delay via μ-Synthesis," IEEE Transactions on Robotics and Automation, Vol 11, No. 1, February 1995.

[4] R. Anderson, M. Spong, "Asymptotic Stability for Force Reflecting Teleoperators with Time Delay," The International Journal of Robotics Research, Vol. 11, April 1992.

[5] G. Niemeyer, J. Slotine, "Stable Adaptive Teleoperation," IEEE Journal of Oceanic Engineering, Vol 16, No. 1, January 1991.

[6] I. Elhajj, N. Xi, Y. Liu, "Real-Time Control of Internet Based Teleoperation with Force Reflection," IEEE International Conference on Robotics and Automation, San Francisco, 2000.

[7] J. H. Ryu, D. S. Kwon, B. Hannaford, "Stable Teleoperation with Time Domain Passivity Control," IEEE International Conference on Robotics and Automation, 2002.

[8] K. Park, W. K. Chung, Y. Youm, "Obtaining Passivity of Micro-Teleoperation Handling Small Inertia Object," IEEE International Conference on Robotics and Automation, 2002.

[9] D. Lawrence, "Stability and Transparency in Bilateral Teleoperation," IEEE Transactions on Robotics and Automation, Vol. 9, No. 5, October 1993.

[10] J. Speich, K. Fite, M. Goldfarb, "A Method for Simultaneously Increasing Transparency and Stability Robustness in Bilateral Telemanipulation," IEEE International Conference on Robotics and Automation, 2000.

[11] J. Speich, M. Goldfarb, "Implementation of Loop-Shaping Compensators to Increase the Transparency Bandwidth of a Scaled Telemanipulation System," IEEE International Conference on Robotics and Automation, 2002.

[12] H. Flemmer, B. Eriksson, J. Wikander, "Control Design for Transparent Teleoperators with Model Parameter Variation," IEEE International Conference on Robotics and Automation, 2002.

[13] T. Brooks, "Telerobotic Response Requirements," IEEE Conference on Systems, Man and Cybernetics, pp. 113-120, 1990.

[14] S. Salcudean, M. Zhu, W. Zhu, K. Hashtrudi-Zaad, "Transparent Bilateral Teleoperation under Position and Rate Control," The International Journal of Robotics Research, Vol. 19, December 2000.

[15] C. Yang, J. Huang, "A Real-Time Synchronization Model and Transport Protocol for Multimedia Applications," Proceedings of IEEE INFOCOM, Canada, June 1994.

[16] T. Znati, R. Simon, B. Field, "A Network-Based Scheme For Synchronization of Multimedia Streams," Proceedings of the Workshop on Perspectives on Multimedia Synchronization, International Conference on Multimedia Computing and Systems, May 15-19, 1995, Washington

[17] T. Little, A. Ghafoor, C. Chen, C. Chang, P. Berra, "Multimedia Synchronization," IEEE Data Engineering Bulletin, Vol. 14, No. 3, Sep. 1991, pp. 26-35.

[18] P. Zarros, M. Lee, T. Saadawi, "Statistical Synchronization Among Participants in Real-Time Multimedia Conference," INFOCOM, pp.912-919, Canada, 1994.

[19] C. Liu, M. Lee, Y. Xie, T. Saadawi, "Multimedia Multipoint Teleconference System with Adaptive Synchronization," IEEE Journal of Selected Area on Communications, vol. 14, no. 7, pp. 1422-1435, Sept. 1996.

[20] Kurose, Jim and Keith Ross, Computer Networking: A Top-Down Approach Featuring the Internet, MA, USA: Addison-Wesley, 2000.

[21] D. Kwon, K. Y. Woo, H. S Cho, "Haptic Control of the Master Hand Controller for a Microsurgical Telerobot System," IEEE International Conference on Robotics and Automation, Vol. 3, pp. 1722-1727, May 1999.

[22] M. Tanimoto, F. Arai, T. Fukuda, and M. Negoro, "Force Display Method to Improve Safety in Teleoperation System for Intravascular Neurosurgery," IEEE International Conference on Robotics and Automation, Vol. 3, pp. 1728-1733, May 1999.

[23] M. Ghodoussi, S. Butner, Y. Wang, "Robotic Surgery –The Transatlantic Case," IEEE International Conference on Robotics and Automation, D.C., 2002.

[24] R. Luo, W. Z. Lee, J. H. Chou, and H. T. Leong, "Tele-Control of Rapid Prototyping Machine Via Internet for Automated Tele-Manufacturing," IEEE International Conference on Robotics and Automation, Vol. 3, pp. 2203-2208, May 1999.

[25] P. G. Backes, K. S. Tso, and G. K. Tharp, "Mars Pathfinder Mission Internet-Based Operations Using WITS," Workshop on Internet Robotics IEEE International Conference on Robotics and Automation, pp. 284-291, May 1999.

[26] L. Hsu, R. Costa, F. Lizarralde, J. Soares, "Passive Arm Based Dynamic Positioning System for Remotely Operated Underwater Vehicles," IEEE International Conference on Robotics and Automation, Vol. 1, pp. 407-412, May 1999.

[27] G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl, "Sensor-Based Space Robotics-ROTEX and Its Telerobotic Features," IEEE Transactions on Robotics and Automation, Vol. 9, No. 5, October 1993.

[28] S. E. Everett, R. V. Dubey, "Model-Based Variable Position Mapping for Telerobotic Assistance in a Cylindrical Environment," IEEE International Conference on Robotics and Automation, Vol. 3, pp. 2197-2202, May 1999.

[29] Robin Murphy, "Human-Robot Interaction in Robot-Assisted Urban Search and Rescue," Workshop on Human-Robot Interaction IEEE International Conference on Robotics and Automation, D.C., 2002.

[30] Gerard McKee, "The development of Internet-Based Laboratory Environments for Teaching Robotics and Artificial Intelligence," International Conference on Robotics and Automation, D.C., 2002.

[31] K. Goldberg, D. Song, Y. Khor, D. Pescovitz, A. Levandowski, J. Himmelstein, J. Shih, A. Ho, E. Paulos, J. Donath, "Collaborative Online Teleoperation with Spatial Dynamic Voting and a Human "Tele-Actor"," International Conference on Robotics and Automation, D.C., 2002.

[32] R. Anderson, M. Spong, "Bilateral Control of Teleoperators with Time Delay," IEEE Transactions on Automatic Control, Vol. 34, No. 5, May 1989.

[33] L. Williams, R. Loftin, H. Aldridge, E. Leiss, W. Bluethmann, "Kinesthetic and Visual Force Display for Telerobotics," IEEE International Conference on Robotics and Automation, D.C., 2002.

[34] S. Ino, S. Shimizu, T. Odagawa, M. Sato, "A Tactile Display for Presenting Quality of Materials by Chnaging the Temperature of Skin Surface," IEEE International Workshop on Robot and Human Communication, 1993.

[35] D. Caldwell, C. Gosney, "Enhanced Tactile Feedback (Tele-Taction) using a Multi-Functional Sensory System," IEEE International Conference on Robotics and Automation, Atlanta, May 1993.

[36] L. Conway, R. Volz, M. Walker, "Teleautonomous Systems: Projecting and Coordinating Intelligent Action at a Distance," IEEE Transactions on Robotics and Automation, Vol. 6, No. 2, April 1990.

[37] G. Hirzinger, K. Landzettel, Ch. Fagerer, "Telerobotics with Large Time Delays-The ROTEX Experience," IEEE/RSJ International Conference on Intelligent Robots and Systems, Munich, Germany, pp. 571-578, September 1994.

[38] A. Bejczy, W. Kim, S. Venema, "The Phantom Robot: Predictive Displays For Teleoperation with Time Delay," IEEE International Conference on Robotics and Automation, pp. 546-551, Cincinnati, May 1990.

[39] R. Riedi, M. Course, V. Ribeiro, R. Baraniuk, "A Multifractal Wavelet Model with Application to TCP Network Traffic," IEEE Transactions on Information Theory (Special Issue on Multiscale Signal Analysis and Modeling), pp. 992-1018, April 1999.

[40] W. Leland, M. Taqqu, W. Willinger, D. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," IEEE/ACM Transactions on Networking, Vol. 2, No. 1, February 1994.

[41] M. Mitsuishi, T. Hori, T. Nagao, "Predictive, Augmented and Transformed Information Display for Time Delay Compensation in Tele-handling/Machining," IEEE International Conference on Robotics and Automation, pp 45-52, 1995.

[42] T. Sheridan, "Space Teleoperation Through Time Delay: Review and Prognosis," IEEE Transactions on Robotics and Automation, Vol 9, No. 5, October 1993.

[43] J. Nilsson, B. Bernhardsson, "Stochastic Analysis and Control of Real-Time Systems with Random Time Delays," Automatica, Vol. 34, pp. 57-64, 1998.

[44] C. Lawn, B. Hannaford, "Performance testing of Passive Communication and Control in Teleoperation with Time Delay," IEEE International Conference on Robotics and Automation, Vol. 3, pp. 776-781, Atlanta, May 1993.

[45] B. Hannaford, W. Kim, "Force Reflection, Shared Control, And Time Delay in Telemanipulation," IEEE International Confernece on Systems, Man, and Cybernetics, Cambridge, November 1989.

[46] J. Klamka, "Observer for Linear Feedback Control of Systems With Distributed delays in Controls and Output," Systems and Control Letters, Vol. 1, 1982.

[47] K. Watanbe, M. Ito, "An Observer for Linear Feedback Control Laws of multivariable Systems with Multiple Delays in Control and Output," Systems and Control Letters, Vol. 1, No. 1, July 1981.

[48] E. Altman, T. Basar, R. Srikant, "Multi-User Rate-Based Flow Control with Action Delays: A Team-Theoretic Approach," Proceedings of the 36^{th} Conference on Decision and Control, San Diego, December 1997.

[49] E. Altman, T. Basar, "Multi-User Rate-Based Flow Control: Distributed Game-theoretic Algorithms," Proceedings of the 36^{th} Conference on Decision and Control, San Diego, December 1997.

[50] U. Madhow, "Dynamic Congestion Control and Error Recovery Over a Heterogeneous Internet," Proceedings of the 36^{th} Conference on Decision and Control, San Diego, December 1997.

[51] Peter Liu, Max Meng, Xiufen Ye, Jason Gu, "End-to-End Delay Boundary Prediction Using Maximum Entropy Principle (MEP) for Internet-Based Teleoperation," IEEE International Conference on Robotics and Automation, D.C., 2002.

[52] Tissaphern Mirfakhrai, Shahram Payandeh, "A Delay Prediction Approach for Teleoperation over the Internet," IEEE International Conference on Robotics and Automation, D.C., 2002.

[53] K. Brady, T. J. Tarn, "Internet-Based Remote Teleoperation," IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 1998.

[54] D. Pai, "ACME, A Telerobotic Measurement Facility for Reality-Based Modelling on the Internet," IROS Workshop on Robots on the Web, Canada, 1998.

[55] R. Simmons, "Xavier: An Autonomous Mobile Robot on the Web," IROS Workshop on Robots on the Web, Canada, 1998.

[56] N. Y. Chong, T. Kotoku, K. Ohba, K. Komoriya, "Remote Coordinated Controls in Multiple Telerobot Cooperation," IEEE International Conference on Robotics and Automation, San Francisco, April 2000.

[57] L. F. Penin, K. Matsumoto, S. Wakabayashi, "Force Reflection for Time-delayed Teleoperation of Space Robots," IEEE International Conference on Robotics and Automation, San Francisco, April 2000.

[58] M. Stein, "Painting on the World Wide Web: The PumaPaint Project," IROS Workshop on Robots on the Web, Canada, 1998.

[59] P. Saucy, F. Mondada, "KhepOnTheWeb: One Year of Access to a Mobile Robot on the Internet," IROS Workshop on Robots on the Web, Canada, 1998.

[60] V. Paxson and S. Floyd, "Wide Area Traffic–The Failure of Poisson Modeling," IEEE/ACM Transactions on Networking, 1995.

[61] J. Beran, R. Sherman, M.S. Taqqu, W. Willinger, "Long-Range-Dependence in Variable-bit-rate Video Traffic," IEEE Transactions on Communication, 1995.

[62] M.E. Crovella, A. Bestavros, "Self-similarity in World Wide Web Traffic: Evidence and Possible Causes," IEEE/ACM Transactions on Networking, 1997.

[63] J. Gao, I. Rubin, "Analysis of Random Access Protocol under Bursty Traffic," Proceedings of IFIP/IEEE International Conference on Management of Multimedia Networks and Services, Chicago, 2001.

[64] T. J. Tarn, N. Xi, A. Bejczy, "Path-Based Approach to Integrated Planning and Control for Robotic Systems," Automatica, Vol. 32, No. 12, pp. 1675-1687, 1996.

[65] J. Tan, N. Xi, T. J. Tarn, "Non-Time Based Tracking Controller for Mobile Robots," IEEE Canadian Conference on Electrical and Computer Engineering, Edmonton Canada, 1999.

[66] N. Xi, T. J. Tarn, "Action Synchronization and Control of Internet Based Telerobotic Systems," IEEE International Conference on Robotics and Automation, Vol.1, pp. 219-224, Detroit, May 1999.

[67] N. Xi, "Event-Based Planning and Control for Robotic Systems," Doctoral Dissertation, Washington University, December 1993.

[68] I. Elhajj, N. Xi, W. K. Fung, Y. H. Liu, Y. Hasegawa, T. Fukuda, "Modeling and Control of Internet Based Cooperative Teleoperation," IEEE International Conferemce on Robotics and Automation, Korea, 2001.

[69] Imad Elhajj, Ning Xi, Wai Keung Fung, Yun hui Liu, Wen J. Li, Tomoyuki Kaga, Toshio Fukuda, "Haptic Information in Internet-Based Teleoperation," IEEE/ASME Transactions on Mechatronics, Vol. 6, No. 3, September 2001.

[70] Imad Elhajj, Jindong Tan, Ning Xi, Wai Keung Fung, Yun Hui Liu, Tomoyuki Kaga, Toshio Fukuda, "Multi-Site Internet-Based Cooperative Control of Robotic Operations," IEEE/RSJ International Conference on Intelligent Robots and Systems, Japan, 2000.

[71] A. Papoulis, "Error Analysis in Sampling Theory," IEEE Proceedings, Vol. 54, July 1966.

[72] M. Unser, "Sampling –50 Years After Shannon," IEEE Proceedings, Vol. 88, April 2000.

[73] N. Hogan, "Multivariable Mechanics of The Neuromuscular System," IEEE Eight Annual Conference of the Engineering in Medicine and Biology Society, 1986.

[74] Y. Zheng, "Human-Robot Coordination for Moving Large Objects," Workshop Note, International Conference on Robotics and Automation, 1997.

[75] T. Milner, "Human Operator Adaptation to Machine Instability," Advances in Robotics, Mechatronics, and Haptic Interfaces, DSC-Vol. 49, ASME 1993.

[76] E. Todosiev, R. Rose, L. Summers, "Human Performance in Single and Two-Axis Tracking Systems," IEEE Transactions on Human Factors in Electronics, Vol. HFE-8, No. 2, June 1967.

[77] G. Bekey, H. Meissinger, R. Rose, "Mathematical Models of Human Operators in Simple Two-Axis Manual Control Systems," IEEE Transactions on Human Factors in Electronics, September 1965.

[78] H. Tan, M. Srinivasan, B. Eberman, B. Cheng, "Human Factors For the Design of Force-Reflecting Haptic Interfaces," Dynamic Systems and Control, DSC-Vol. 55-1, 1994.

[79] J. Tan, N. Xi, "Hybrid System Design for Singularityless Task Level Robot Controllers," IEEE International Conference on Robotics and Automation, 2000.

[80] Canudas de Wit, Carlos, Bruno Siciliano and Georges Bastin, Theory of Robot Control, England: Springer-Verlag, 1996.

[81] Slotine, Jean-Jacques and Weiping Li, Applied Nonlinear Control, NJ, USA: Prentice-Hall, 1991.

[82] Khalil, Hassan, Nonlinear Systems, NJ, USA: Prentice-Hall, 1996.

[83] Tadao Murata, "Petri Nets: Properties, Analysis and Applications," Proceedings of the IEEE, Vol. 77, No. 4, April 1989.

[84] A. Moro, H. Yu, G. Kelleher, "Advanced Scheduling Methodologies for Flexible Manufacturing Systems using Petri Nets and Heuristic Search," IEEE International Conf. on Robotics and Automation, San Francisco, pp.2398-2403, Arpil 2000.

[85] Baccelli, F., G. Cohen, G. Olsder and J. Quadrat, Synchronization and Linearity, England: John Wiley and Sons, 1992.

[86] Peterson, James, Petri Net Theory and the Modeling of Systems, NJ, USA: Prentice-Hall, 1981.

[87] M. Song, "Integration of Task Scheduling, Sensing, Planning and Control in a Robotic Manufacturing Work-Cell," Doctoral Dissertation, Washington University, August 1997.

[88] M. Rezai, M.R. Ito and P.D. Lawrence, "Modelling and Simulation of Hybrid Control Systems by Global Petri Nets," IEEE International Symposium On Circuits and Systems, Seattle, 1995.

[89] M. Gotesman, N. LopezBenitez, "Petri Netbased Modeling of Hybrid Dynamic Systems," IEEE Conference on Emerging Technologies and Factory Automation, Hawaii, November 1996.

[90] G. Nenninger, V. Krebs, "Modeling and Analysis of Hybrid Systems: A New Approach Integrating Petri Nets and Differential Equations," IEEE Joint Workshop on Parallel and Distributed Real-Time Systems, 1997.

[91] Tadao Murata, Zhehui Wu, "Fair Relation and Modified Synchronic Distances in a Petri Net," Journal of the Franklin Institute, Vol. 320, No,2, pp. 63-82, August 1985.

[92] J. Moody, K. Yamalidou, M. Lemmon, P. Antsaklis, "Feedback Control of Petri Nets Based on Place Invariants," Proceedings of the 33rd Conference on Decision and Control, FL. Dec. 1994.

[93] L. E. Holloway, B. H. Krogh, "Controlled Petri Nets: A Tutorial Survey," Lecture Notes in Computer Science, v. 199, Eleventh International Conference on Analysis and Control, Discrete Event Systems, June 1994.

[94] J. Moody, P. Antsaklis, M. Lemmon, "Automated Design of a Petri Net Feedback Controller for a Robotic Assembly Cell," IEEE Symposium on Emerging Technologies and Factory Automation, Vol. 2, pages 117-128. October 1995.

[95] M. Uchiyama, "A Unified Approach to Load Sharing, Motion Decomposing and Force Sensing of Dual Arm Robots," Robotic Research: The Fifth International Symposium, MIT Press, 1990.

[96] Y. F. Zheng, J.Y. S. Luh, "Optimal Load Distribution for Two Industrial Robots Handling a Single Object," Int. Conf. on Robotics and Auto., 1988.

[97] K. Kosuge, T. Oosumi, H. Seki, "Decentralized Control of Multiple Manipulators Handling an Object in Coordination Based on Impedance Control of Each Arm," IEEE/RSJ Int. Conference on Intelligent Robots and Systems, France, 1997.

[98] Z. Wang, E. Nakano, T. Matsukawa, "Realizing Cooperative Object Manipulation using Multiple Behavior-Based Robots," IEEE/RSJ International Conf. on Intelligent Robots and Systems, 1996.

[99] N. Y. Chong, T. Kotoku, K. Ohba, K. Komoriya, K. Tanie, J. Oaki, H. Hashimoto, F. Ozaki, K. Maeda, N. Matsuhira, "A Collaborative Multi-site Teleoperation over an ISDN," Mechatronics Journal, in press.

[100] N. K. Chong, T. Kotoku, K. Ohba, K. Tanie, "Virtual Repulsive Force Field Guided Coordination for Multi-telerobot Collaboration," IEEE International Conf. on Robotics and Auto., Korea 2001.

[101] A. Kheddar, C. Tzafestas, P. Coiffet, T. Kotoku, S. Kawabata, K. Iwamoto, K. Tanie, I. Mazon, C. Laugier, R. Chellali, "Parallel Multi-Robots Long Distance Teleoperation," IEEE International Conference on Advanced Robotics, California, 1997.

[102] T. Suzuki, T. Fujii, K. Yokota, H. Asama, H. Kaetsu, I. Endo, "Teleoperation of Multiple Robots through the Internet," IEEE Inter. Workshop on Robot and Human Communication, 1996.

[103] David Williams, Oussama Khatib, "The Virtual Linkage: A Model for Internal Forces in Multi-Grasp Manipulation," IEEE International Conf. on Robotics and Auto., Atlanta, Georgia, 1993.

[104] Amer alYahmadi, T. C. Hsia, "Internal Force-Based Impedance Control of Dual-arm Manipulation of Flexible Objects," IEEE Inter. Conf. on Robotics and Automation, San Francisco 2000.

[105] A. Ramadorai, T. J. Tarn, A. Bejczy, N. Xi, "Task-Driven Control of Multi-Arm Systems," IEEE Transactions on Control Systems Technology, Vol. 2, No. 3, September 1994.

[106] Imad Elhajj, Ning Xi, BooHeon Song, Meng-Meng Yu, Wang Tai Lo, Yun Hui Liu, "Transparency and Synchronization in Supermedia Enhanced Internet-Based Teleoperation," Inter. Conf. on Robotics and Auto., D.C. 2002.

[107] Nomad XRDEV Software Manual. Release 1.0. Nomadic Technologies, Inc., 1999.

[108] H. Kazerooni, "Human-Robot Interaction via the Transfer of Power and Information Signals," IEEE Transactions on Systems, Man and Cybernatics, 1990.

[109] Ning Xi, T. J. Tarn, "Heterogeneous Function-Based Human/Robot Cooperations," IEEE International Conference on Robotics and Automation, Belgium, May 1998.

[110] J. Luh, Y. Zheng, "Constrained Relations Between Two Coordinated Industrial Robots for Motion Control," International Journal Robot. Res., pp.60-70, 1987.

[111] M. Raibert, J. Craig, "Hybrid Position/Force Control of Manipulators," Transactions of ASME, 1981.

[112] Ning Xi, T.J. Tarn, A. Bejczy, "Intelligent Planning and Control for Multirobot Coordination: An Event-Based Approach," IEEE Transactions on Robotics and Automation, Vol.12, 1996.

[113] K. Ohba, S. Kawabata, N. Y. Chong, K. Komoriya, T. Matsumaru, N. Matsuhira, K. Takase, K. Tanie, "Remote Collaboration Through Time Delay in Multiple Teleoperation," IEEE/RSJ International Conference on Intelligent Robots and Systems, Korea, 1999.

[114] J. Tan, N. Xi, "Integrated Sensing and Control of Mobile Manipulators," IEEE/RSJ International Conference on Intelligent Robots and Systems, Hawaii, December 2001.

[115] T. J. Tarn, N. Xi, C. Guo, A. Bejczy, "Human/machine Sharing Control for Telerobotic Systems," Intelligent Robots and Systems, 1995.

[116] T. Balch, R.C. Arkin, "Behavior-based formation control for multirobot teams," IEEE Trans. Robotics and Automation, Vol. 14, pp. 926-939, 1998.

[117] Imad Elhajj, Jindong Tan, Yu Sun and Ning Xi, "Supermedia Enhanced Human/Machine Cooperative Control of Robot Formations," IEEE/RSJ International Conference on Intelligent Robots and Systems, Switzerland, 2002.

[118] H. Kobayashi, H. Nakamura, "A Scaled Teleoperation," IEEE International Workshop on Robot and Human Communication, 1992.

[119] K. Kaneko, H. Tokashiki, K. Tanie, K. Komoriya, "Macro-Micro Bilateral Teleoperation Based on Operational Force Feedforward," IEEE/RSJ international Conference on Intelligent Robots and Systems, Canada, 1998.

[120] King W. C. Lai, Carmen K. M. Fung, Wen J. Li, Imad Elhajj, Ning Xi, "Transmission of Multimedia Information on Micro Environment via Internet," International Conference of the IEEE Industrial Electronics Society, Nagoya Japan, 2000.

[121] Antony W. T. Ho, Wen J. Li, Imad Elhajj, Ning Xi, "Development of a Bone Reaming System Using Micro Sensors for Internet Force-Feedback Control," Workshop on Service Automation and Robotics, Hong Kong, June 2000.

[122] Carmen K. M. Fung, King W. C. Lai, Wen J. Li, Yunhui Liu, Imad Elhajj, and Ning Xi, "Sensing and Action in a Micro Environment via Internet," International Conference on Information Society in the 21st Century, Aizu, Japan, November 2000.

[123] Terrence Fong, Charles Thorpe, Charles Baur, "Advanced Interfaces for Vehicle Teleoperation: Collaborative Control, Sensor Fusion Displays and Remote Driving Tools," Autonomous Robots, pp. 75-85, 2001.

[124] Nikitas Sgouros, Stelios Gerogiannakis, "Integrating WAP-based Wireless Devices in Robot Teleoperation Environments," IEEE International Conference on Robotics and Automation, D.C., 2002.

[125] Pablo d'Angelo, Peter Corke, "Using a WAP Phone as Robot Interface," IEEE International Conference on Robotics and Automation, D.C., 2002.