Multi-Site Internet-Based Cooperative Control of Robotic Operations *

I. Elhajj, J. Tan, N. Xi
Dept. of Electrical
and Computer Engineering
Michigan State University
East Lansing, MI 48824, U.S.A.

W. K. Fung, Y. H. Liu
Dept. of Mechanical
and Automation Engineering
Chinese University of Hong Kong
Shatin; N.T., Hong Kong

T. Kaga, Y. Hasegawa, T. Fukuda
Center for Cooperative Research
in Advanced Science and Technology
Nagoya University
Nagoya 464-8603, Japan

Abstract

The e-world, also known as the Internet, has added a new dimension to many of the traditional concepts in industrial applications and everyday life. Many words encountered nowadays are prefixed by the letter “e” to reflect their close relation to the electronic world. Therefore, it was not surprising when researchers and businesses started developing “e-services”, which are simply services provided via the Internet. The use of robots has dramatically expanded the potential of e-services. Now individuals with particular expertise can perform highly accurate and fairly complicated tasks remotely via the Internet. This increase in the human reachability is faced by several obstacles.

Reliable and efficient robot facilitated services via the Internet face several challenges. These range from human-computer interfacing and overcoming random time delay to task synchronization and human-robot interaction. These limitations intensify when multi-operators in multi-sites are involved. This paper provides new theoretical and experimental results on these challenges. Specifically, multi-site cooperative control of an Internet based mobile manipulator is presented. The two main characteristics of this system are Internet based real-time closed loop control and coordinated operation. In addition, it is shown that despite random time delay the stability and synchronization of the system were achieved using event-based control.

1 Introduction

Most of the work done in cooperation in robotics can be divided into two kinds: human-robot, and multi-robot. Human-robot cooperation occurs when a hu-
2 Non-Time Based Multi-Site Robotic System Model

Event-based control was first introduced by N. Xi [7]. The key idea of event-based control is the use of a reference, \( s \), rather than time \( t \). As a consequence of using this reference, the model of all the components in such systems will be a function of the event, \( s \), rather than time \( t \). The general structure of a multi-operator multi-robot collaborative control with force reflection system is shown in Fig.1 and all the terms are explained in Table 1. This model shows that the force fed back to each operator can be a result of the other operators’ desired velocities or the robots’ interaction with the environment or even a combination of both. The designer chooses a particular force scenario depending on the particular task setup.

![Diagram](image)

**Figure 1:** Detailed block diagram of a general multi-operator multi-robot collaborative control system.

<table>
<thead>
<tr>
<th>Block</th>
<th>Our Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human Operator</td>
<td>( F_1 \in \mathbb{R}^3 ): Applied force</td>
</tr>
<tr>
<td></td>
<td>( X_{mi} \in \mathbb{R}^3 ): Joystick pos.</td>
</tr>
<tr>
<td>Master</td>
<td>( V_{mi} \in \mathbb{R}^3 ): Velocity desired</td>
</tr>
<tr>
<td>Slave</td>
<td>( V_{si} \in \mathbb{R}^3 ): Actual velocity</td>
</tr>
<tr>
<td></td>
<td>( V_{fi} \in \mathbb{R}^3 ): Feedback</td>
</tr>
<tr>
<td>Environment</td>
<td>( V_{envi} \in \mathbb{R}^3 ): Real or Virtual contact</td>
</tr>
</tbody>
</table>

**Table 1:** Explanation of the various variables in Fig.1.

The special case presented in this paper, where a mobile manipulator was used, is modeled in Fig.2. Each block will be discussed in detail and all terms are explained in Table 2.

**Human Operator:** This is the most difficult to model, but a spring-like behavior may be assumed, as shown in [10] and as used in several instances in the literature [9]. The human operator also provides stability to the coupled human-machine system since the human can compensate for certain machine instabilities [11]. Once the operators feel a force, they will generate a new joystick position according to the following:

\[
X_m(s) = \frac{F_p(s - 1)}{K_m} \quad X_p(s) = \frac{F_m(s - 1)}{K_p}
\]

(1)

where \( K_m \) and \( K_p \) are scaling constants, \( s \) is the event and \( s \in \mathbb{R} \). \( X_m, X_p, F_m \) and \( F_p \) are:

\[
\begin{pmatrix} X_m(s) \\ X_p(s) \end{pmatrix} = \begin{pmatrix} X_{ma}(s) \\ X_{mp}(s) \end{pmatrix}
\]

(2)

\[
\begin{pmatrix} F_m(s) \\ F_p(s) \end{pmatrix} = \begin{pmatrix} F_{ma}(s) \\ F_{mp}(s) \end{pmatrix}
\]

(3)

As Table 1 shows, \( F_m(s) \) and \( F_p(s) \) are the applied forces, i.e. the forces that the operators feel. Thus, the \( x \) and \( y \) components are due to the force fed back and the additional force required to move the joystick to the new location. Since force is not fed back in the \( \theta \) direction, this component is just a result of getting the joystick to the new location. As seen in eq.1, \( X_m(s) \) and \( X_p(s) \) are related to \( F_p(s - 1) \) and \( F_m(s - 1) \), so \( X_m(s) \) and \( X_p(s) \) at event \( s \) are generated by the previous force at event \( s - 1 \). This results in an event-based system where each event is triggered by the previous one.

![Diagram](image)

**Figure 2:** Block diagram of the multi-operator mobile manipulator teleoperation system implemented.

**Master (Joystick):** The dynamics of the joysticks are:

\[
M_mV_{mm}(s + 1) = F_p(s) + F_{V_m}(s)
\]

(4)

\[
M_pV_{mp}(s + 1) = F_m(s) + F_{V_p}(s)
\]

(5)
<table>
<thead>
<tr>
<th>Block</th>
<th>Our Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human Operator</td>
<td>$F_{m}, F_{p} \in \mathbb{R}^3$: Applied force $X_m, X_p \in \mathbb{R}^3$: Joystick pos.</td>
</tr>
<tr>
<td>Master</td>
<td>$V_m, V_p \in \mathbb{R}^3$: Velocity desired</td>
</tr>
<tr>
<td>Communication Link</td>
<td>$V_m \in \mathbb{R}^3$: Velocity set</td>
</tr>
<tr>
<td>Slave</td>
<td>$V \in \mathbb{R}^3$: Actual velocity</td>
</tr>
<tr>
<td>Environment</td>
<td>$V_{in} \in \mathbb{R}^3$: Virtual contact</td>
</tr>
</tbody>
</table>

Table 2: Explanation of the various variables in Fig 2.

\[
F_{m}(s) = C_m V_m(s) \quad (6) \\
F_{p}(s) = C_p V_p(s) \quad (7)
\]

where $M_m$ and $M_p$ are the masses of the joysticks handles, $V_{mm}$ and $V_{mp}$ are velocity of joysticks movement, and $F_m$ and $F_p$ are as described earlier. $F_{m}$ and $F_{p}$ 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. 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

\[
V_m(s) = K_{m1} X_m(s) \quad V_p(s) = K_{p1} X_p(s) \quad (8)
\]

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:

\[
V_m(s) = \begin{bmatrix} V_{mx}(s) \\ V_{my}(s) \\ V_{m\theta}(s) \end{bmatrix} \quad V_p(s) = \begin{bmatrix} V_{px}(s) \\ V_{py}(s) \\ V_{p\theta}(s) \end{bmatrix} \quad (9)
\]

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. Since the advance of time does not affect the system and only the advance of the event $s$ does, the system will remain stable when the connection is lost and will resume action only after the connection is re-established. This makes the system very robust since no initialization or synchronization is required.

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 $f(d)$ that would give a velocity value $V_m(s)$. $V_{in}(s)$ is subtracted from the desired velocity $V_m(s)$ to give the velocity set for the robot $V_s(s)$,

\[
V_s(s) = V_m(s) - V_{in}(s) \quad (10)
\]

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

\[
V_s(s) = \begin{bmatrix} V_{sx}(s) \\ V_{sy}(s) \\ V_{sm}(s) \end{bmatrix} \quad V_{in}(s) = \begin{bmatrix} V_{inx}(s) \\ V_{iny}(s) \\ 0 \end{bmatrix} \quad (11)
\]

Slave1 (Mobile robot): Once the robot receives $V_s(s)$, it will be commanded to move with that velocity, but it will actually move at velocity $V_a(s)$. The dynamics of the robot are described by the following equations:

\[
M_s \ddot{V_a}(s) = F_s(s) + \tau_a(s) \quad (12) \\
\tau_a(s) = -\gamma V_a(s) + K V_{err}(s) - \alpha_f F_s(s) \\
V_{err}(s) = V_s(s) - V_a(s) \quad (13) \\
V_a(s) = \begin{bmatrix} V_{ax}(s) \\ V_{ay}(s) \\ V_{a\theta}(s) \end{bmatrix} \quad (14)
\]

Here $\gamma$, $K$ and $\alpha_f$ are constants, $M_s$ is mass of the robot, and $F_s$ is the actual environment forces if any, usually assumed very small. $\tau_a$ and $V_{err}$ are robot internal controller terms.

Slave2 (PUMA manipulator): The dynamic model for the robot arm can be written as

\[
D(q)\ddot{q} + c(q, \dot{q}) + g(q) = u \quad (15)
\]

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

Let $y \in \mathbb{R}^6$ 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, Attitude and Tool angles). The Dynamic model of the arm can be simplified to: $\ddot{Y} = u$

### 3 Non-Time Based Control: Stability and Synchronization

Time delay in communication links has several effects on the stability and synchronization of teleoperation systems. These effects are intensified when teleoperation with force feedback is considered and even more so when multi-operators at multi-sites are involved. An event-based approach was used to ensure stability and synchronization of real-time control of
Internet based teleoperation with force reflection [6]. The objective of this paper is to further develop this approach and generalize it to multi-operator collaborative control of multiple robots with force reflection. To ensure the stability of such systems the reference, $s$, has to satisfy a condition that follows from the theorem below [8]:

**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) nondecreasing function of time $t$, then the system is (asymptotically) stable with respect to the new action reference $s$.

The only assumption needed is that the robot is a stable system, which means that the original robot dynamic system (without remote human operator) is asymptotically stable with $t$ as its action reference. This would allow the use of Theorem 1 and prove 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) nondecreasing function of time $t$.

For the particular case of stability in teleoperation, taking $s$ to be the number of commands that the system executes satisfies the above condition and results in a stable system. The advantage of this approach is that stability is proven independent of the human model or the statistics of time-delay. In addition, this theorem can be applied to multi-robot control if each robot controller is stable when considered without remote human operation.

To describe the motivation behind the selection of $s$ as the reference and to demonstrate the synchronization of the system, the control and communication within the system’s different components will be detailed. Fig. 3 depicts the system which consists of two operators at different sites and a mobile manipulator. One operator controls the mobile base and the other controls the manipulator.

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 operator 2, who is controlling the manipulator, generates the desired velocity vector $V_p$ and sends it to the manipulator for execution. Meanwhile, the PC of operator 1, 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 veloc-

![Figure 3: Hardware structure of the system.](image)

ity 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 do 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 that will be discussed in sec. 4. 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 operator 2 and operator 1 consecutively. This implies that the operators are aware of each other’s intentions, which makes collaboration more efficient and safe.

This control algorithm implies that the two operators and the two robots are event-synchronized. Event-synchronization is defined here as follows:

**Definition 1** An event-synchronized system is one in which all the entities in the system are referencing the same common event, $s$, at any point in time.

This definition specifies that two different entities in a system can not be at different events; in a closed loop system, the feedback obtained has to correspond to the most up-to-date status of the plant being controlled. As for the collaborative system presented here, this implies that the operators are each feeling the most recent intention of the other.

### 4 Implementation

The implementation can be divided into two parts, hardware and software. As shown in Fig. 3, the system consists of various operating systems and configurations. So the problem of interconnection had
to be studied carefully. The joysticks used are programmable Microsoft SideWinder Force Feedback Pro, with 3 degrees of freedom. The mobile robot is a Nomadic XR4000 and the manipulator is a PUMA 350.

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

Motion 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. The obstacle avoidance algorithm first checks the different sensors on the robot, which are used to detect the distance to the closest object in the direction of motion. If any sensor detects an object closer than $d_c$, which is a pre-defined programmable critical distance, the motion is stopped. Otherwise the closest distance, $d$, is used to calculate the velocity to be set. Velocity is set to $V_s$, which is given in eq.10, where $V_{sp} = f(d)V_m$. Once the motion 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 motion 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 motion 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 singularityless hybrid motion controller is used to avoid the singularities of the robot arm [12].

Motion client: This client sends commands $V_m$ to the motion 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 operator from an overhead camera.

This implementation allows both operators to feel the other robot’s commands. But this does not eliminate the need for having direct force feedback between the robot and its operator as the one used in [6].

5 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. However, the system performance is stable and synchronized as will be shown in the experiments presented. 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 Fig.4, 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. The top row of Fig.4 shows a plot of time versus $s$, the event. It is clear that $s$ is a non-decreasing function of time, which was crucial for proving 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 rotational 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 $s$. Fast response is clear from the sharp decrease in the error between the velocities of the two robots.

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 desti-
nation 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.

6 Conclusions

This paper has presented multi-site cooperative control of an Internet-based mobile manipulator with the novel feature of force feedback. The force feedback to each of the operators conveys the intentions of the other operator, which improves performance and provides efficient coordination.

Event-based control has been used to ensure stability and synchronization. This was theoretically proven and experimentally demonstrated. In addition, the new concept of event-based synchronization was introduced. The benefits of such synchronization in teleoperation were presented and demonstrated in the teleoperation system discussed. These qualities demonstrate the potential that “e-services” have, where a link between the virtual and real world can be created. E-services don’t have to be limited to electronic applications anymore, but can be expanded to offer physical services. So the future can include services such as: remote physical assistance (elderly and disabled assistance), tele-medicine (remote checkup and surgery) and remote clean-up (hazard and explosive material manipulation).

References


