Prioritized Data Transfer for a Bilateral Robot Control via Real-Time Network System

Yutaka Uchimura* Member
Nobuyuki Yamasaki** Non-member
Kouhei Ohnishi*** Member

This paper describes a prioritization scheme for robot-control-related data that are transmitted via a real-time network system. The network system is constructed by a device named Responsive Processor and the network is applied to a bilateral robot system. The robot system is driven by a controller based on impedance matching theory. Due to a limitation of the network capacity, the priority order of the robot-control data must be determined. We analyzed a control system with time delay and assigned the priority order considering the robustness against time delay. To evaluate the priority assignment, we conducted experiments on a network based bilateral robot. The experimental results verified both the prioritization scheme and the order of the priority assignment.

Keywords: Telerobotics, real time systems, delay effects, force control

1. Introduction

Network-based motion control such as a tele-operation provides substantial interaction with the real world. The tele-operation system allows an operator to move, touch and manipulate a remote environment via a network. For these applications, real-time performance of the data communication is a high-priority requirement. Particularly when the data communication path comprises a feedback loop, it directly affects the performance of the control system. This well-known problem has motivated a great deal of research.

One of the well-studied approaches is a method based on robust control theory, which deals with the variant time delay factor as an unmodeled uncertainty. Another approach is to transform the system into a passive system by applying the scattering transformation to the communication channel with time delay. These methods are very powerful; however, they require an upper bound for the time delay. It implies that the time delay of the network should ideally be deterministic from the view of control theory.

As many researchers have revealed, some network systems are not deterministic due to their media access methods. Such systems do not guarantee an upper bound for the time delay on the network transmission. For example, the Ethernet, one of the most popular and widely used networks, uses the CSMA/CD arbitration mechanism, which leads to a random time delay during transmission. This indefinite delay makes the network non-deterministic.

To eliminate this drawback, a novel network processor named 'Responsive Processor' was developed. The Responsive Processor provides a genuine real-time network with two physically separated paths. One path is designed for the data transfer with short delay and the other is designed for a high throughput transfer.

When implementing the processor to an application such as the robot control, robot-control-related data are urgent and require a short delay. However, the capacity (packet size) of the short-delay path is limited and thus one must prioritize the data to assign the available slots. The issue is how to determine the priority order of the data.

In this paper, we describe a prioritization scheme for the control-related data transmitted via the network that considers the time delay factor. The robustness of each control-related data against time delay is evaluated, and then the priority order is assigned. Based on this priority scheme, the real-time network by the Responsive Processor was implemented on a bilateral robot system, which was driven by a controller based on impedance matching theory.

The following section describes the Responsive Processor and the structure of the message handler. In Section III, we describe a controller design that is based on impedance matching. In Section IV, we analyze the robustness against time delay and evaluate the priority assignment. Finally, we show experimental results in Section V.

2. The Responsive Processor

The Responsive Processor is designed for parallel and distributed real-time control applications. Fig. 1 shows the Responsive Processor on PCI form factor. It is equipped with a SPARC core MPU, I/O functions and a network interface in an ASIC chip. Among those functions, the network interface, hereafter 'Responsive Link', has the key role in the real-time data communication.

The most significant feature of the Responsive Link is that it has two separate communication links. One of them, 'event link', is designed for short-latency packet transmission. The other is 'data link', which is for high-bandwidth data communication. Because these two links are physically separated,
the packet, which goes on the event link, should never be delayed no matter how busy the data link is. This feature is the principal design scheme to maintain real-time performance of the network system. There often exists a trade-off on choosing the event link or the data link. A general solution is to evaluate the priority of the transmitting data, and then assign the event link for the higher priority data. More details on this are discussed later.

A packet used on the event link is called an event packet, and a packet on the data link is called a data packet. The size of both packets is fixed such that an event packet is 16 bytes and a data packet is 64 bytes. An event packet consists of a 4-byte header, an 8-byte payload and a 4-byte trailer. A data packet consists of a 4-byte header, a 56-byte payload and a 4-byte trailer.

To implement the motion control application, we developed a message handler that manages the transmission of the packet between the Responsive Processor and a host PC. The host PC acquires the sensor data and drives the actuators by generating control signals. In the design of the message handler, the principal scheme to manage priority-driven task scheduling is also maintained. To pass data between the Responsive Processor and the host PC, mailbox and DMA transfer are mainly used. The mailbox generates an interrupt signal, thus it is used to pass the urgent event packet. On the other hand, DMA is mainly used for high throughput data transfer such as data packets.

On the host PC side, priority-driven scheduling was implemented by using RT-Linux. We assigned a higher priority to the thread that handles event packets than data packets.

3. Controller Design for a Bilateral Robot Based on Impedance Matching

We applied the Responsive Processor-based real-time network system to a force-reflecting bilateral robot. Fig. 2 shows the configuration of the robot system. The system is comprised of two robot manipulators, host computers, and the Responsive Processors. Each manipulator has two degrees of freedom and a force sensor is installed on the tip of the arm. When an operator maneuvers the master manipulator, the slave manipulator draws the same motion. Once the remote manipulator touches an object, the operator feels it through the sense of touch.

For a bilateral tele-operation, the ideal situation is to have the operator feel the remote object without being conscious of the intervening machine. This is called 'transparency', and many studies have been seeking a better solution to achieve it. A typical approach is the impedance matching method, which uses an analogy between a dynamics model and an electrical circuit. The four-channel controller, which has a generalized architectures to achieve transparency, is also based on impedance matching.

Fig. 3 depicts the two-port model, which is commonly used in circuit theory and is suitable to describe the impedance matching in bilateral tele-operation. In the figure, Z, F and X represent impedance, force and position respectively. Subscripts m, s, e and h denote master, slave, environment and human operator, respectively.

The dynamics of the master manipulator is expressed as $F_m = Z_mX_m$, whereas the slave is $F_s = Z_sX_s$. Therefore, if the impedance matching conditions, $Z_m = Z_e$ and $Z_s = Z_h$, are satisfied, perfect transparency is achieved. With transparency, the impedance of the master is exactly the same as the environment so the operator feels as if he or she is directly handling the remote object.

The design method proposed by Nishioka is a practical solution based on the impedance matching to accomplish the transparency. The structure of the controller is essentially a hybrid control of force and position. The controller equivalently works as a compliance control whose compliance is adjusted to match the impedance of the environment (the remote object or the operator). It is achieved using a hybrid combination ratio that prioritizes the value of the position control and the force control.

Fig. 4 shows the block diagram of the control system. The control law is described in (1).

$$u_\text{m} = -\gamma K_f (f_s + f_m) + (1 - \gamma)(K_p + K_s)(x_s - x_m)$$

$$\gamma = \frac{x_s^\text{T}}{x_s^\text{T} + M_\text{u}^{-1} x_s} \quad \text{(1)}$$
where $\gamma$ is the hybrid combination ratio, $u_m$ is the command value to the master manipulator, $f_m$ and $f_s$ are the force of master and slave, $x_m$ and $x_s$ are the position; and $K_f$, $K_p$, and $K_s$ are the feedback gains for force, position, and velocity, respectively. $M_e$ is the equivalent mass of the remote object, which can be designed arbitrary. Equation (1) describes the control law of the master side only; however we assume that the slave manipulator has identical structure and dynamics so the slave side control has the same structure as the master.

The hybrid combination ratio $\gamma$ is an important part of this design method, because it equivalently adjusts the impedance of the master manipulator to the slave environment. Thus, this ratio greatly affects on the dynamic structure of the system.

4. Prioritization of the State Variables

Because the event link can transmit a packet with less delay than data link, all the control-related data should be sent via the event packet. However, its payload size is too small to carry all the data and thus only some of the control data can be transmitted. In particular, the control of a manipulator with two degrees of freedom requires sending data such as two sets of position, velocity and force as well as the hybrid ratio, whereas the payload size of the event packet is 8 bytes. Therefore only two of 32-bit floating-point data can be sent in one packet. To expand the capacity, we tried converting 32-bit floating-point data (IEEE 754 format) to 16-(or 14-) bit floating-point data, which consists of 1 bit for the sign, 4 bits for the exponent and 11 (or 9) bits for the significand. Despite the effort, the payload size was still too large. Therefore we need to choose which data to be transmitted via the event link.

Among the control-related data, we chose the hybrid ratio $\gamma$ in (1) to be sent via the event packet because the ratio represents the dynamical change of the environment. This ratio tells the system whether or not the slave manipulator has touched an object and thus it greatly affects the dynamic structure of the system. The values of the hybrid ratio are packed into two of the 14-bit floating point data as shown in Fig. 5.

Moreover, four bits are allocated to the command signals such as start, stop and the most important emergency stop. Consequently, the rest of the available spaces is limited to two of 16-bit floating point data elements. In fact, the velocity can be calculated from the position; therefore, the spaces are assigned for either position or force.

Due to such a constraint, we need to prioritize the state variables based on robustness against time delay. The higher prioritized state variables are to be transmitted via the event link. The rest of the data is to be sent via the data link.

For the purpose of determining the order of priority of the state values, we set up an analysis model that has the impedance-matching-based controller that we described in the previous section. Using this model, we evaluated the robustness of each state variable against time delay. Fig. 6 shows the model for the analysis.

In the figure, $G_m(s)$ and $G_s(s)$ are the models of master and slave manipulators, $Z_o(s)$ and $Z_e(s)$ are the impedances of the operator and the environment, $L_f(s)$ is a low pass filter for the force sensor, $K_p$ is the position feedback gain, $K_s$ is the velocity feedback gain and $K_f$ is the force feedback gain. Each gain is assumed to be constant value. The hybrid ratio $\gamma$ can vary continuously between 0 and 1. The time delay is modeled as $e^{-\tau D}$, where $D$ is the magnitude of the time delay. The time delay causes a phase delay whose magnitude increases as the frequency increases, whereas the gain maintains a constant value. $D(s)$ is a pseudo-derivative for obtaining the velocity from the position. Because the velocity is calculated using the position value, we assumed that both of the position and the velocity are delayed by the same amount.

The robustness against time delay is evaluated as follows:

1. Design the controller gain ($K_p$, $K_s$, $K_f$) assuming no time delays.
2. Obtain the open loop transfer function from $u_1$ to $y_1$ in Fig. 6 (as the position feedback loop).
3. Obtain the open loop transfer function from $u_2$ to $y_2$ in Fig. 6 (as the force feedback loop).
4. Calculate the phase margin and the gain crossover frequency for each feedback loop.
5. Add the phase delay (due to the time delay) and evaluate the phase margin at the gain crossover frequency.
6. The robustness against time delay is then measured as the total phase margin.

Note that adding the phase delay (due to the time delay) to the open loop transfer function corresponds to adding the time delay to the system.

We analyzed the phase delay of the position control and the force control by using various combinations of numerical values. An example is the followings:
Feedback gains were chosen so that the position feedback loop and the force feedback loop had the same phase margins without time delay. These numerical values are a typical example for the model system. We tested various combinations of values and obtained virtually the same qualitative results as the above.

The robustness was examined on the bode plot by changing the value of the hybrid ratio $\gamma$ from 0 to 1. Fig. 7 and Fig. 8 show the bode plot for the case when $\gamma = 0.5$. Fig. 7 shows the case for the force data are delayed and Fig. 8 shows the analogous bode plot for the position. In both figures, the dashed line shows the phase plot of the transfer function (from $u_k$ to $y_i$ where $i = 1, 2$) and the solid line shows the case when the data (force or position) are delayed. Fig. 9 shows the bode plot in the case when $\gamma = 1$ and the force data are delayed. Fig. 10 shows the case when $\gamma = 0$ and the position data are delayed.

Comparing the phase delays at the gain crossover frequency, the delay of position control drops down under $-180$ degrees as shown in Fig. 8 and Fig.10, which means that the system becomes unstable. Conversely, the phase delay for the force control stays above $-180$ degrees (positive phase margin) as shown in Fig. 7 and Fig. 9. These results are caused by the fact that the gain crossover frequency for the position control is higher than that for the force control. The phase delay due to the time delay is larger at higher frequencies; therefore, the position feedback is affected by the time delay more than the force feedback. Consequently, the state variables associated with position feedback are less robust against time delay.

From these numerical results, we judged that the position data should be assigned higher priority than the force data, and thus should be sent via the event link.

5. Performance Evaluation by Experiments

We conducted experiments to evaluate the effect due to time delay and examine the validity of the priority assignment. Fig. 11 shows the overview of the experimental setup. The manipulator in front is the master manipulator that the operator maneuvers. The slave manipulator is in the back. The objects to touch are a sponge block or a hard paper box. These two manipulators are connected through the real-time network by the Responsive Processors. The sampling time of the control system was 1 msec.

Before showing the results with time delay, we describe some results without time delay as reference. Fig. 12 and Fig. 13 show the results for two different objects without time delay. Fig. 12 is the case of a hard object (hard paper box) and Fig. 13 is that of a soft object (sponge block). In each figure, the upper plot shows the force response in the direction of the z-axis (the gravity direction). For convenience, the sign of the master manipulator is reversed. The lower graph shows the position of the manipulator's tip in the direction of the z-axis. Solid lines indicate response of the master side.
and dashed lines show the slave side. In both cases, the position and force-tracking performances are more than sufficient. From these results we find that control algorithm functioned as desired and the impedance matching between the master and the slave are satisfactory accomplished.

The force response against the hard object and the soft object show similar responses, in which the magnitude reaches approximately 15 N. On the other hand, the peak-to-peak position changes are much larger in the case of the sponge block than that of the paper box. This result indicates that the operator feels the sponge is apparently softer through the larger deformation.

Next, let us show the results with time delay. In fact, due to a limitation in the number of network devices, the network configuration is point-to-point connection with two Responsive Processor boards. In this configuration, even the data link does not produce any time delay over the control sampling time (1 msec). Therefore, we emulated the time delay by the software implementation using a FIFO buffer, which stores the current data in a buffer memory then outputs past (delayed) data.

Assuming that the some of the variables are transmitted via the event link and the rest of them are sent via the data link with time delay, we tested various combinations of variables and time delays. The results shown here are the cases when time delay was set to 20 msec.

Fig. 14 shows the response against a paper box when the force data were delayed for 20 msec whereas the position data were not. It shows a stable response, although slight fluctuations can be observed at the peak response though. However, the feel of touch was almost as good as the case without delay.

Fig. 15 shows the response against a paper box when the position data were delayed for 20 msec, whereas the force data were not delayed. The result shows an oscillating response near around the peak value, which is induced by an unstable contact. This result shows that the delay of position value greatly affects the control system.

Judging from these results, the priority order of position first then force is appropriate. In fact, the bandwidth of the position feedback is wider than that of the force feedback. This is because the bandwidth of the low-pass filter is so narrow that the magnitude of feedback gain is limited. If a less noisy force sensor is developed and we can enlarge the cut-off frequency of the low-pass filter, the priority order may be affected.
6. Conclusion

We proposed a prioritization scheme for the control-related data that is transmitted via a real-time network system by the Responsive Processor. Due to the limitation of the capacity of short-latency path, we assigned the priority order for state variables.

We analyzed the impedance-matching-based bilateral robot control system with and without time delay and determined the priority order considering the robustness of control-related data against time delay. The prioritization scheme was applied to a bilateral robot system connected via the Responsive Processor. Experiments with a bilateral robot system showed that the priority assignment was effective and thus supported the prioritization scheme.

Acknowledgment

This study was performed through Special Coordination Funds of the Ministry of Education, Culture, Sports, Science and Technology of the Japanese Government.

(Manuscript received June 22, 2004)

References


Yutaka Uchimura (Member) received the B.E. and M.E. degree in electrical engineering from Keio University, Yokohama, Japan, in 1989, and 1991, respectively. Since 1991, he has been with Kajima Technical Research Institute, Tokyo, Japan, where he is currently a senior researcher. Since 2001, he has been a Ph.D. candidate at Keio University's Graduate School program. His main research areas are real time systems, computer networks, distributed systems, and robotics.

Nobuyuki Yamasaki (Non-member) received the B.S. degree in physics, and the M.S. and Ph.D. degrees in computer science from Keio University, Yokohama, Japan, in 1991, 1993, and 1996, respectively. Since 1998, he has been with the faculty of information and computer science, Keio University. He is now an associate professor. His research interests include real-time processing, real-time communication, distributed real-time systems, computer architecture, RT-OS, robotics, and VLSI design.

Kouhei Ohnishi (Member) received the B.E., M.E., and Ph.D. degrees in electrical engineering from the University of Tokyo, Tokyo, Japan, in 1975, 1977, and 1980, respectively. Since 1980, he has been with Keio University, Yokohama, Japan. His research interests include mechatronics, motion control, and robotics.