Implementation and Control of Generic Internet-Based Telerobotic Systems

Sudath R. Munasinghe, Ju-Jang Lee, Yuji Ishida, Naruto Egashira, and Masatoshi Nakamura

Abstract: This paper describes the design and implementation techniques of a generic Internet-based telerobotic system. The purpose is to make available the precious technical information and hands-on experience of the authors for the research community. The paper is based on the telerobotics system that the authors recently implemented between KAIST, Korea and Saga University, Japan. In its current functionality, two control modes: high-level supervisory control and low-level supervisory control are explored demonstrating their merits and demerits.

Keywords: Internet-based teleoperation, real-time control, supervisory control, telerobotics.

1. INTRODUCTION

Telerobotics is becoming an increasingly important research and application area. Recently, telerobotics and teleoperations have been introduced in a wide range of applications. Ranging from space and orbital [1] tasks, teleoperations has found its potential in underseas [2], medical [3], welfare [4], rescue [5], and entertainment [6] applications. Nonetheless, telerobotics research is limited to a very few institutions. Furthermore, it is not in the main stream of robotics research mainly due to the fact that it requires a wide range of multi-disciplinary expertise to construct and maintain a telerobotics system. It also seems that the technical details of telerobotics system designs have not been properly documented and made widely available for the research community. As such, it has become an exhaustive task to design and implement a telerobotics system.

In this work, we present a detailed account on the design and implementation of a generic Internet-based telerobotics system. The design details may help novice researchers to quickly grasp the key essentials required to implement their own telerobotics systems. The generic telerobotics system described in this paper is based on the client-server concept with the client as the remote operator, and the server as the local controller of the telerobot. We have adopted supervisory control in that the remote operator sends pertinent information to the local controller, and the local controller carries out trajectory planning and robot control independently. Two-level supervisory control modes are described in this generic design; high-level supervisory control and low-level supervisory control.

(a) *High-level supervisory control mode*: The remote operator sends only motion and control parameters to the local controller. The local controller plans the entire trajectory and controls the robot independently.

(b) *Low-level supervisory control mode*: The remote operator sends consecutive positions in an incremental order. The local controller plans the incremental motion trajectories using uniform velocity and sampling interval, and controls the robot.

By nature, high-level supervisory control requires that an advanced trajectory planner such as those proposed in [7] and [8] reside in the local controller, whereas low-level supervisory control can be implemented with a very simple trajectory planner such as uniform velocity interpolation. Based on these two supervisory control modes, we have designed and implemented an Internet-based telerobotics system between the Korea Advanced Institute of Science and Technology (KAIST) and Saga University, Japan. Almost all design features and details of this telerobotics system are presented in this paper. The two supervisory control modes

Manuscript received August 1, 2003; revised March 22, 2004; accepted March 23, 2004. Recommended by Editorial Board member Wankyun Chung under the direction of Editor Keum-Shik Hong.

Sudath R. Munasinghe and Ju-jang Lee are with the Department of Electrical Engineering and Computer Science, KAIST, 373-1 Guseong-dong, Yuseong-gu, Dejeon 305-701, Korea (e-mail: rohan@odyssey.kaist.ac.kr, jjlee@ee.kaist.ac.kr).

Yuji Ishida and Masatoshi Nakamura are with the Department of Advanced Systems Control Engineering, Saga University, 1 Honjomachi, Saga 840-8502, Japan (e-mail: {ishida, nakamura}@cntl.ee.saga-u.ac.jp).

Naruto Egashira is with the Department of Control and Information Systems Engineering, Kurume National College of Technology, 1-1-1 Kumorino Kurume-City, Fukuoka 830-8555, Japan (e-mail: naruto@kurume-nct.ac.jp).

have been tested on this telerobotics testbed, verifying successful design and functionality. The high-level control mode provides better results, but, it can be employed only in static environments. Low-level supervisory control can be used to realize maneuvers, which are unknown a priori, by slightly compromising the performance level. Both these control strategies are useful in telerobotics applications. Directions for further developments and research are also discussed.

2. SUPERVISORY CONTROL AND SYSTEM DESIGN

The most fundamental and crucial issue in telerobotics is the delay in signal transmission between the remote operator and the telerobot controller [9]. In direct telemanipulation, where the remote operator is a part of the control loop, the feedback control loop encompasses both sides of the telerobotics system. The remote operator receives feedback from the local controller, and determines the control action to be sent to the local controller over the transmission network. When the telerobotics system spans over a long geographical distance, such as ground-based space telerobotics, the transmission delay can be significantly lengthy, and the delayed feedback in the direct teleoperation can force the system into an unstable state. The only means to preserve stability is to slow down the operating speed significantly.

Supervisory control [9] has overshadowed most direct teleoperations due to the fact that it can be used to implement a local control loop. This is a possibility in most robotic manipulations, where the maneuvers can be planned and executed by the local controller independently, with the help of a trajectory planner. As such, the remote operator only has to oversee the performance of the telerobot while sending necessary instructions to the local controller. Due to the local implementation of the control loop, there is no feedback time delay, and no problem with stability.

2.1. High-level supervisory control strategy

For simple motions and manipulations in static environments, the high-level supervisory control mode is well-suited. Fig. 1 illustrates the telerobotics system implemented between KAIST and Saga University, Japan based on the high-level supervisory control mode. In this mode, the remote operator sends motion and control parameters to the local

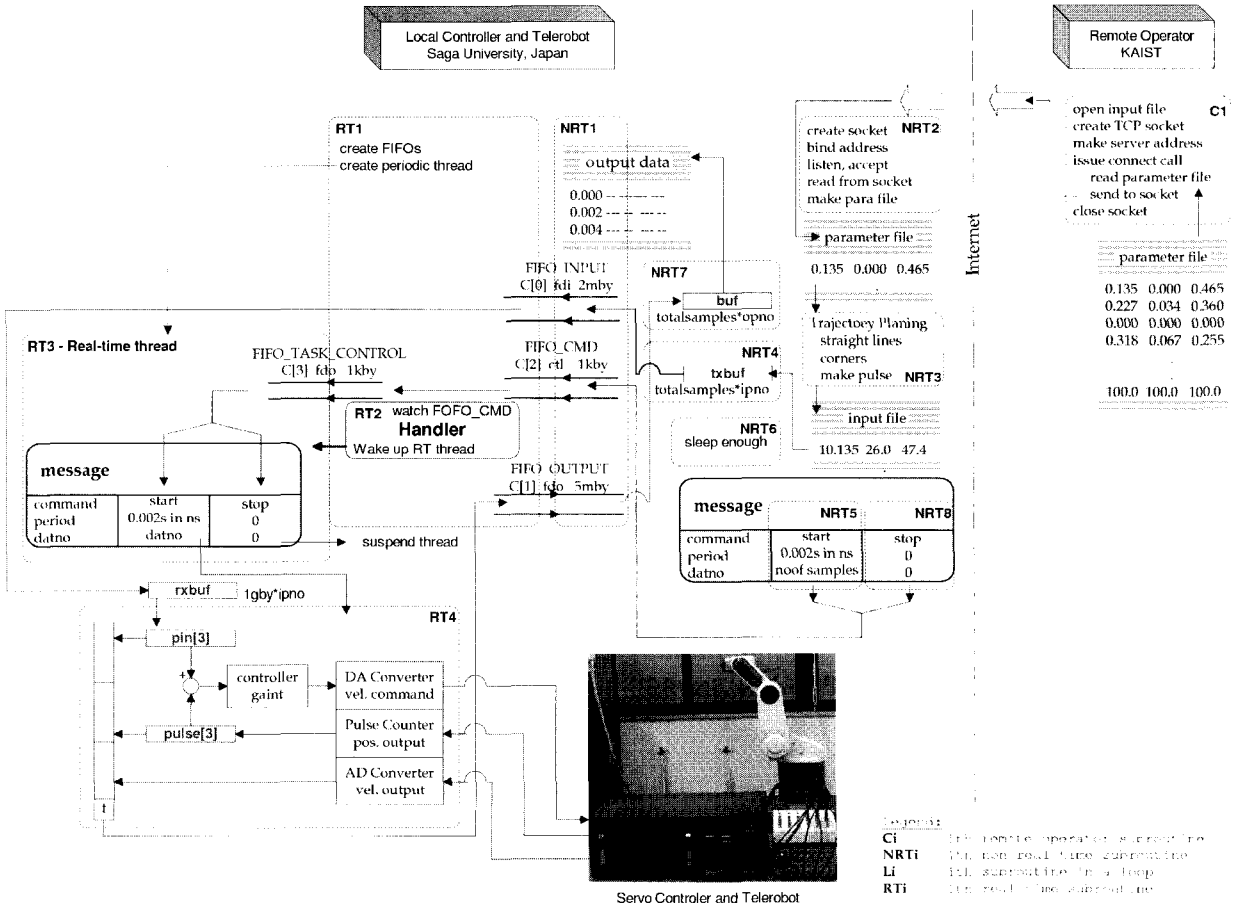


Fig. 1. Telerobotics test bed between KAIST, Korea and Saga University, Japan in high-level supervisory control mode.

controller. The local controller uses the trajectory planner proposed in [8], and controls the robot autonomously.

Remote Operator:

As illustrated in Fig. 1, the remote operator resides in KAIST, Korea, and the local controller and telerobot reside in Saga University, Japan. The two sides communicate through TCP sockets [10] through the Internet. The remote operator reads motion parameters from a resident file, and sends those parameters to the local controller in Saga University (**C1**).

Local Controller:

The local controller operates a non real-time process (**NRT1~NRT8**). It reads motion and control parameters through a TCP socket, and writes them onto a file (**NRT2**). The entire motion trajectory is planned by an advanced trajectory planner (**NRT3**), which eventually creates the input data file. This pulse file contains the input data sequences of the planned motion of the telerobot. The pulse data is then transferred to a serial buffer whose size is determined by the number of controlled joints of the telerobot, and number of input data samples. The input data is then written to the first-in-first-out buffer, FIFO-INPUT (**NRT4**). The message structure is then written with the "start" command, together with a sampling interval used by the trajectory planner, and number of input samples in the planned trajectory. This message structure is placed in FIFO-CMD (**NRT5**). From this point, the non real-time (**NRTs**) process waits, allowing time for the real-time (**RT**) process to carry out telerobot control (**NRT6**).

The local controller also operates a real-time process (**RT1~RT4**), which is programmed using Pthreads [11]. A handler is created for FIFO_CMD so that any command written to FIFO_CMD is immediately noticed by the handler (**RT2**). When the handler sees a new command in FIFO_CMD, it copies it onto FIFO_TASK_CONTROL, and immediately sends a wake-up signal to the real-time periodic thread that runs the control loop of the telerobot (**RT3**). Being woken up by the handler, the real-time periodic thread reads the command from

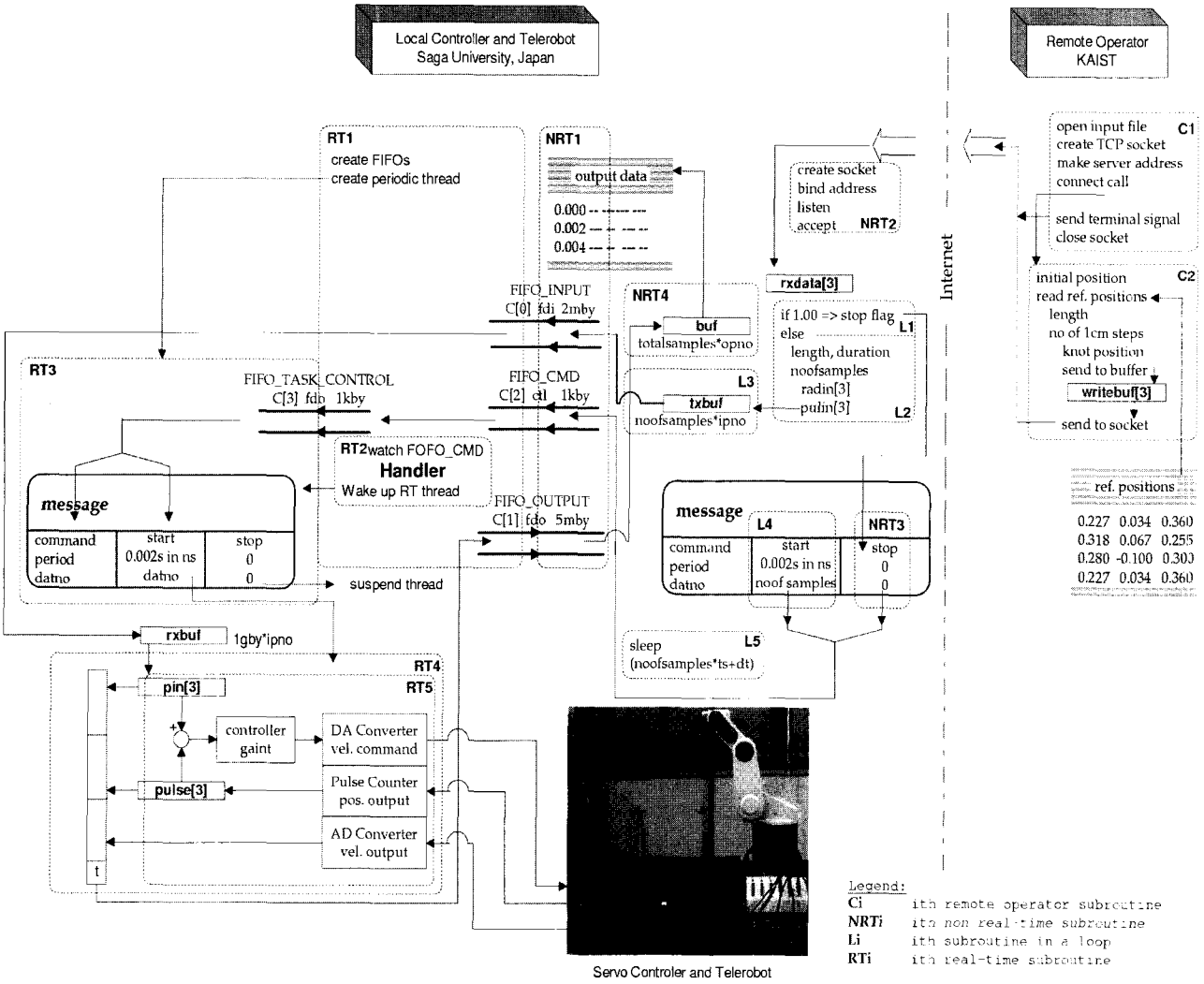


Fig. 2. Telerobotic systems implemented between KAIST and Saga University based in low-level supervisory control strategy.

FIFO_TASK_CONTROL to its message structure. Once it locates the "start" command, it then reads the message structure member variables to determine the sampling interval and number of input data samples for the planned motion. The input data is then read from FIFO_INPUT to a serial buffer named rxbuf, which is large enough to hold all the input data of the planned trajectory. By this time, the real-time periodic thread is ready to carry out the planned motion of the telerobot. The control loop (**RT4**) is located within the real-time periodic thread, and is operated by executing the feedback control loop with the input data in the serial buffer, rxbuf. In our implementation, we use PI control to regulate joint error. The number of control loop operations is already specified in the message structure. In every control loop operation, joint positions of the telerobot are read from the servo controller. A pulse counter provides position feedback whereas a digital-to-analog (DA) converter provides velocity feedback. The output data is immediately sent to FIFO_OUTPUT.

The non real-time process then issues the "stop" command to FIFO_CMD (**NRT8**). The handler reads it and immediately places it onto FIFO_TASK_CONTROL, waking up (or interrupting) the running real-time thread (**RT3**). Once the real-time thread sees the "stop" command it suspends the control loop, and the real-time process terminates. The non real-time process continues for a while, and reads the output data from the FIFO_OUTPUT to a serial buffer (**NRT7**), which is long enough to hold the total amount of output data. Output is then written to a data file, following correct row-column configuration.

2.2. Low-level supervisory control strategy

The low-level supervisory control strategy is illustrated in Fig. 2. This design looks similar to the previous high-level supervisory control design, with a few major changes. On the remote operator side, reference inputs are read repetitively from a file. For each reference position, it operates a loop (**C2**) within which it segments the individual motions into a set of equi-distant positions, and sends them to the local controller through a TCP socket. The local controller also runs a non real-time loop (**L1~L4**) within which it reads the incremental positions, distance between them, duration for the incremental motion, and number of input samples in each incremental motion (**L1** and **L2**). The input data for the incremental motion of the telerobot is sent to a serial buffer and then placed onto FIFO_INPUT (**L3**). This incremental motion is planned assuming uniform end-effector speed. Then, the "start" command is written onto the message structure, together with the number of input data samples for the incremental motion. The message structure is placed onto the FIFO_CMD, and the non-real time process waits allowing time for the real-time process to control the telerobot along the incremental motion.

The real-time process has the handler (RT2) and the periodic real-time thread (RT3) operating in the same way as described in the high-level supervisory control design. One exception, however, is that the control loop has one additional mode (RT5). This control mode operates during the idling time between two consecutive incremental motions. It uses the last input data as the input for the robot joint controllers, and stabilizes the robot at that point until the next incremental position input data appears. However, it does not write output to the FIFO_OUTPUT. Both real-time and non real-time processes terminate essentially the same way as in the previous design.

2.3. Implementation

The real-time processes were programmed using real-time Linux. The remote operator was implemented on RedHat® Linux v7.3 with kernel version 2.4.20-13.8, whereas the local telerobotic controller was implemented on Debian GNU Linux 2.2 with real-time kernel version 2.2.19-rtl. In high-level supervisory control, the remote operator sends motion and control parameters as:

- Reference end-effector positions, P1-P2-P3-P4
- Assigned end-effector velocity, and
- Joint acceleration limits

whereas the entire trajectory is planned autonomously by the local controller in **NRT3** according to the algorithm shown in Fig. 3.

The trajectory planning algorithm is extensively described in [8]. Only a brief account is given here. As depicted in Fig. 3, motion and control parameters are provided to the trajectory planner.

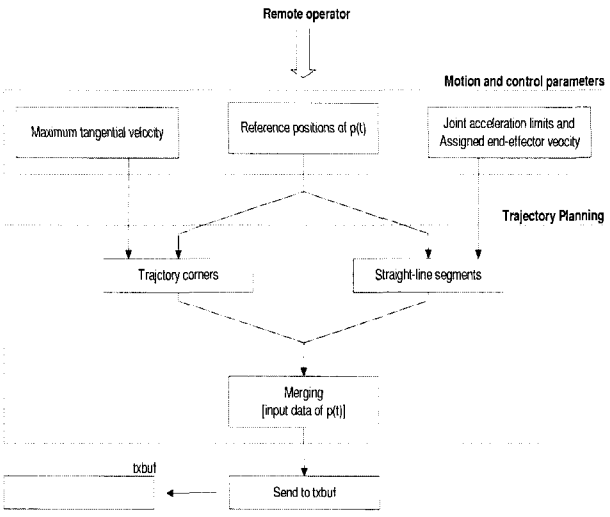


Fig. 3. Autonomous trajectory planner for high-level supervisory control.

According to the reference positions, corners and straight-lines can be identified. Corners are planned with the maximum tangential velocity v_t^m . The maximum tangential velocity is determined to avoid joint torque saturation as the end-effector moves along a trajectory corner. Joint torques are likely to saturate at trajectory corners, therefore it is necessary to reduce the end-effector speed substantially as described in [12]. Straight-lines are planned using maximum joint acceleration, provided end-effector is not over-speeding, otherwise, uniform end-effector speed is used to plan the straight line. Once straight line segments and corners have been planned, they are merged together.

For Internet security reasons, the local controller has not been assigned a global IP address. Instead, we used a virtual private network (VPN) connection between the two sides of the system. A simple visual feedback has been implemented with Microsoft® NetMeeting, and it operates in isolation with the control loop. It only allows the remote operator to oversee the motion of the Telerobot.

3. EXPERIMENTAL RESULTS

3.1. High-level supervisory control

In the high-level supervisory control mode, the reference end-effector positions P1(0.227, 0.034, 0.360), P2(0.318, 0.067, 0.255), P3(0.280, -0.100, 0.300), P4=P1, and assigned end-effector velocity, v_a =0.03[m/s] were the motion parameters sent by the remote operator.

The results of the experiment are shown in Fig. 4 in which the realized trajectories coincide with the corresponding planned trajectories making them indistinguishable in that resolution. The reference end-effector positions P1~P4 are labeled at the top. The end-effector trajectory in three dimensional Cartesian spaces, as shown in Fig. 5, indicate that the two rounded corners at P2 and P3 are the result of the trajectory planner [8].

3.2. Low-level supervisory control

In the low level supervisory control experiment, the following reference trajectory was used: Q1(0.227, 0.034,0.360), Q2(0.318, 0.067, 0.255), Q3(0.280, 0.034, 0.360), and Q4=Q1. Distance of the incremental end-effector motion was set to 1cm.

The incremental position generator (**C2** in Fig. 2) generates 14 increments between Q1~Q2, 17 increments between Q2~Q3, and 15 increments between Q3~Q4, all using the uniform end-effector velocity. These incremental positions are sent to the local controller of the Telerobot, where trajectory segments are planned between incremental positions using uniform speed and sampling interval. Results of the joint and Cartesian positions are shown in Fig.

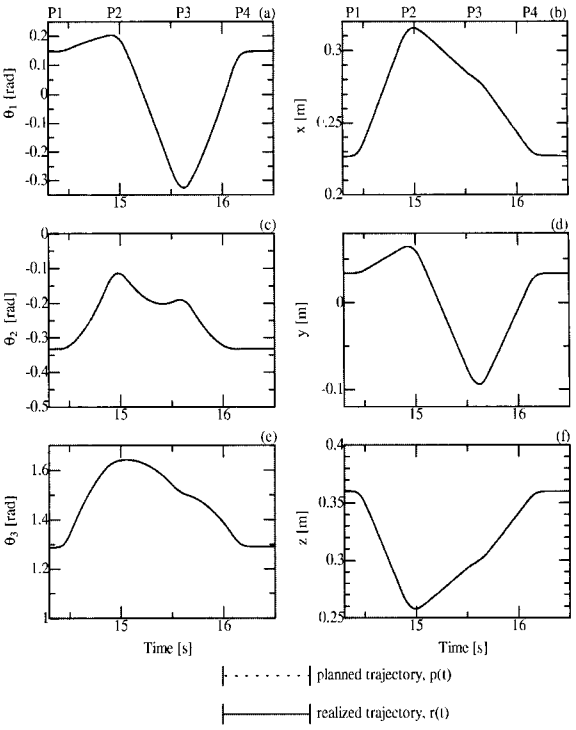


Fig. 4. Joint and Cartesian position results of the high-level supervisory control.

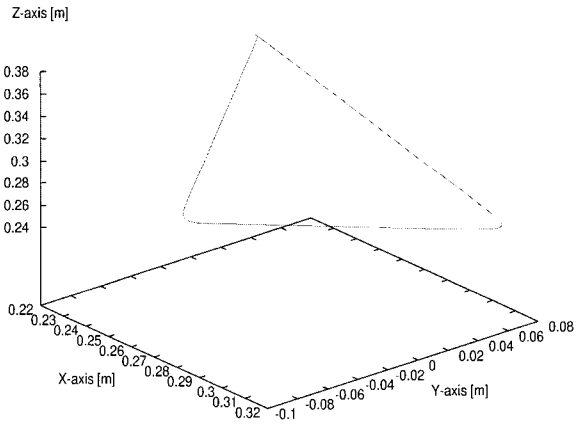


Fig. 5. End-effector trajectory performed under high-level supervisory control.

6. The end-effector motion in a 3D workspace is shown in Fig. 7.

3.3. Evaluation and discussion

The end-effector motion performance in both high-level and low-level supervisory control modes are accurate and comparable (Figs. 5 and 7). The planned end-effector trajectories in these two experiments almost perfectly coincide with the two followed end-effector trajectories, and therefore cannot be distinguished in the figures. High-level supervisory control exhibits much better velocity performance compared to low-level supervisory control (Fig. 8).

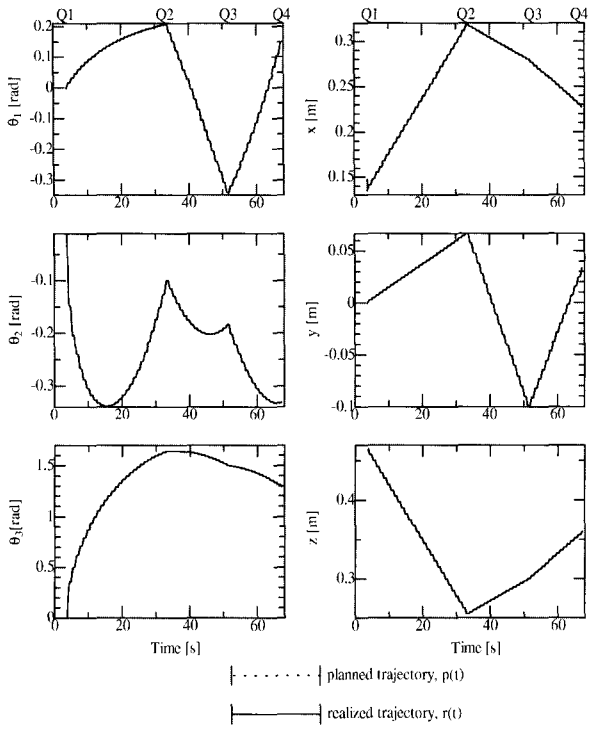


Fig. 6. Joint and Cartesian position results of the low-level supervisory control.

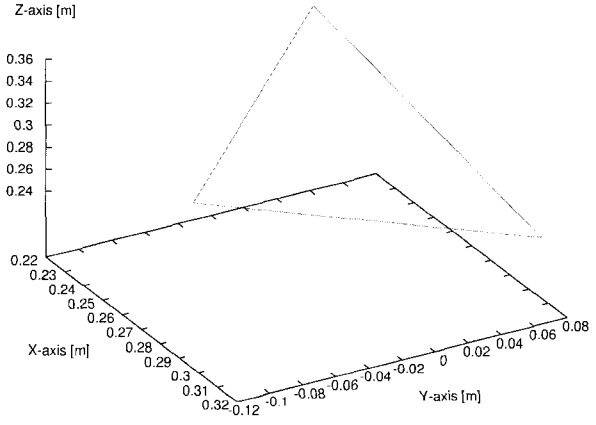


Fig. 7. End-effector motion under low-level supervisory control.

Low-level supervisory control uses 0.02 [m/s] as the desired end-effector velocity. As labeled in Fig. 8(b), A1 shows an intensive spike as input velocity jumps from zero to 0.02[m/s]. The velocity fluctuation due to move-and-wait motion is indicated by B1 in the same figure. In high-level control, an advanced trajectory planner [7,8] constructs the entire end-effector trajectory so that the resulting manipulator motion becomes much smoother compared to that of low-level control.

Supervisory control starts to mimic direct control when shorter incremental motions are used. However, deep down in the control architecture, it still is supervisory control, as the control loop is locally

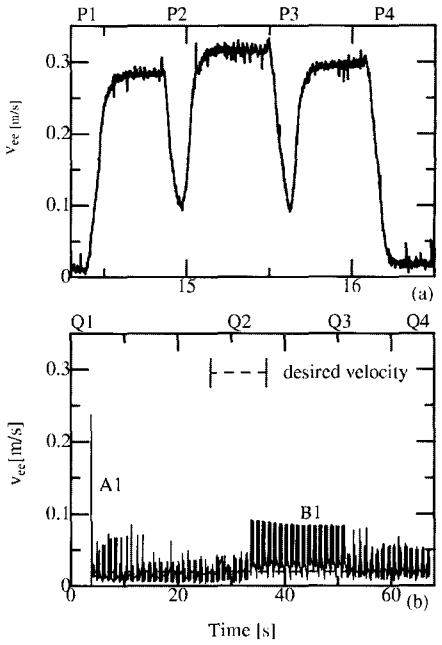


Fig. 8. End-effector velocity: under (a) high-level supervisory control, and (b) low-level supervisory control.

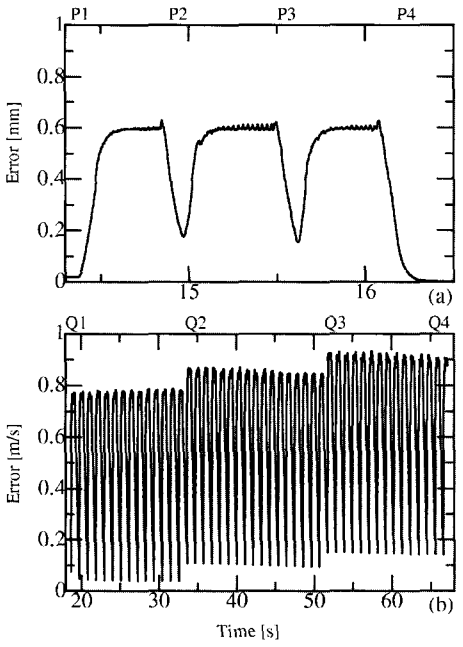


Fig. 9. Error of the end-effector trajectory: (a) with high-level supervisory control, and (b) low-level supervisory control.

implemented. As expected, high-level supervisory control has a lower magnitude and smoother variation in end-effector position error as shown in Fig. 9.
Low-level control demonstrates a constant repeti-tion in every incremental move of the end-effector. Both error profiles show that the error magnitude is proportional to the end-effector velocity. It gives experimental proof of how effective the velocity con-

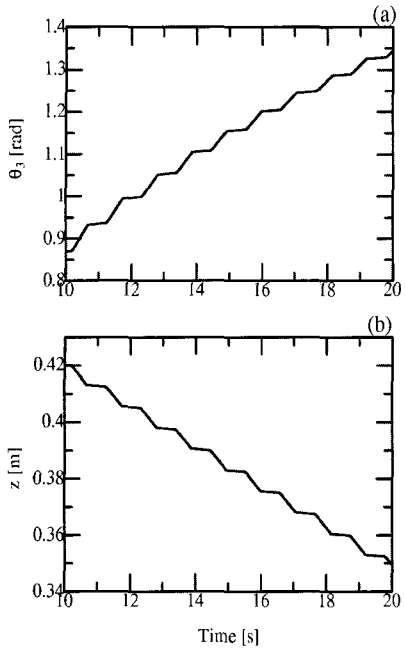


Fig. 10. Incremental motion in joints and Cartesian space.

trol is in determining the final end-effector performance. The move-and-wait nature of the incremental motions under low-level control is visible in Fig. 6, and more elaborated in Fig. 10 by enlarging a small time slice of Fig. 6.

The waiting time after every incremental move is one of the reasons why low-level control operation takes an excessively long time, about 48[s] to complete a comparable physical motion, which is performed under high-level control in about 16[s].

High-level supervision always brings better performance, yet it can only be applied for relatively simple tasks, in static environments. On the contrary, low-level supervisory control can be applied for complex tasks in dynamic environments, where other control strategies would simply fail.

The virtual private network (VPN) connection between KAIST, Korea and Saga University, Japan, recorded round trip time delay as 15ms~350ms. Due to the very nature of supervisory control, time delay has no effect on the system stability.

4. CONCLUSION AND FURTHER DEVELOPMENTS

Design features of an Internet-based telerobotics system were discussed in this paper. Two control modes: high-level supervisory control and low-level supervisory control, and their corresponding telerobotics system designs have also been presented with experimental results. Novice telerobotics researchers and engineers would find these designs, implementations, and results as important guidelines for them to design their own telerobotics systems. The merits and demerits of high-level and low-level control strategies in telerobotics have also been discussed.

As for further development, a hand device could be attached at the remote operator side, to generate remote operator commands in a more human interactive manner. The waiting time in low-level supervisory control should be minimized in order to make it more continuous in motion.

REFERENCES

[1] G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl, "Sensor-based space robotics—ROTEX and its telerobotic features," *IEEE Trans. on Robot. and Automat.*, vol. 9, no. 5, pp. 649-663, 1993.

[2] P. G. Neckes and M. K. Long, "Local-remote telerobotics for underwater vehicles," *Proc. Symp. on Autonomous Underwater Vehicle Technology*, pp. 11-15, 1992.

[3] A. Rovetta. R. Sala, W. Xia, and A. Togno, "Remote control in telerobotic surgery," *IEEE Trans. on Syst., Man and Cyber. Part A*, vol. 26, no. 4, pp. 438-444, 1996.

[4] S. R. Munasinghe and M. Nakamura, "Teleoperation of welfare robotic systems by motion planning considering assigned velocity and acceleration limit," *Intl. Journal of Human-Friendly Welfare Robotic Systems*, vol. 3, no. 2, pp. 23-31, 2002.

[5] A. Rovetta, "FRIEND robot, space telerobot for rescue and recovery of astronauts," *IEE/RSJ Intl. Workshop on Intelligent Robots and Systems*, vol. 3, pp. 1663-1668, 1991.

[6] K. Goldberg, "Mercury Project: A feasibility study for internet robots," *IEEE Robotics and Automation Magazine*, vol. 7, no. 1, pp. 35-40, 2000.

[7] S. R. Munasinghe, M. Nakamura, S. Goto, and N. Kyura, "Optimum contouring of industrial robot arms under assigned velocity and torque constaints," *IEEE Trans. on Syst., Man, and Cyber.*, vol. 31, no. 2, pp. 159-167, 2001.

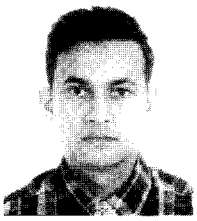
[8] S. R. Munasinghe, M. Nakamura, S. Goto, and N. Kyura, "Trajectory planning for industrial robot manipulators considering assigned velocity and allowance under joint acceleration limit," *Intl. Journal of Control, Automation and Systems*, vol. 1, no.1, pp. 68-75, 2003.

[9] T. B. Sheridan, *Telerobotics, Automation and Human Supervisory Control*, MIT Press, Cam., MA, 1992.

[10] W. A. Gay, *Linux Socket Programming*, Que Inc, Indianapolis, 2000.

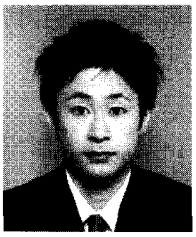
[11] B. Nichols, D. Buttler, and J. P. Farrel, *Pthreads Programming*, O'Reilly & Associates, Inc., Sebastipol, CA, 1996.

[12] S. R. Munasinghe and M. Nakamura, "Determination of maximum tangential velocity at trajectory corners in robot manipulator operation under torque constraint," *SICE Intl. Conference*, MA17-2, pp. 1170-1175, 2002.
[13] S. R. Munasinghe, M. Nakamura, S. Aoki, S. Goto, and N. Kyura, "High speed precise control of robot arms with assigned speed under torque constraint by trajectory generation in joint co-ordinates," *Proc. IEEE Conf. on Syst., Man, and Cyber.*, vol. 2, pp. 854-857, 1999.

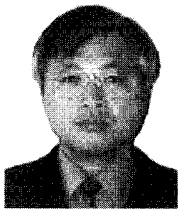


Sudath R. Munasinghe was born in Colombo, Sri Lanka, in 1970. He received the B.Sc. degree in electrical engineering from the University of Moratuwa, Sri Lanka in 1996. He received the M.Sc. and Ph.D. degrees in electrical engineering form Saga University, Japan in 1999 and 2003, respectively. From May 1996 to June 1997 he was a Junior Staff Member in the Department of Electrical Engineering, University of Moratuwa. From October 1999 to April 2000, he was a Senior Staff Member in the Department of Computer Science and Engineering, University of Moratuwa. Currently, he is a Postdoctoral Researcher in the Department of Electrical Engineering and Computer Science, Korea Institute of Science and Technology. His research interests include manipulator control, intelligent control, humans in the loop control, force-position hybrid control, teleoperation, and space robotics.

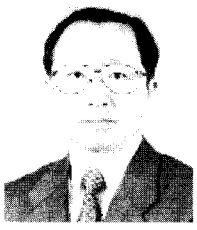
Ju-Jang Lee was born in Seoul, Korea, in 1948. He received the B.S. and M.S. degrees in electrical engineering from Seoul National University, Korea in 1973 and 1977, respectively, and the Ph.D. degree in electrical engineering from the University of Wisconsin, U.S.A, in 1984. From 1977 to 1978, he was a Research Engineer at the Korean Electric Research and Testing Institute, Korea. From 1978 to 1979, he was a Design and Processing Engineer at the G.T.E. Automatic Electric Co., U.S.A. In 1983, he participated as the Project Engineer for a brief time in the Research and Development Department of the Wisconsin Electric Power Co., U.S.A. Following that, he joined the Department of Electrical Engineering at KAIST in 1984, where he is currently a Professor. In 1987, he was a Visiting Professor at the Robotics Laboratory of Imperial College of Science and Technology, U.K. From 1991 to 1992, he was a Visiting Scientist at the Robotics of Carnegie Mellon University, U.S.A. His current research interests include intelligent and variable structure control of mobile robots, evolutionary emotional robots, service robots for the disabled, space flexible manipulators, intelligent transportation systems, and power system stabilizers.
Prof. Ju-Jang Lee is a Senior Member of IEEE, IEEE Robotics and Automation Society, IEEE Evolutionary Computation Society, and IEEE Industrial Electronics Society. He is a Vice President of ICASE in Korea and a Director of SICE in Japan. He is also an Editor of the Artificial Life and Robotics and Machine Intelligence and Robot Control.



Yuji Ishida was born in 1979 in Kumamoto Japan. He received the B.Sc. degree in electrical and electronics Engineering from Saga University, Japan, in 2002. Since then, he has been pursuing the M.Sc. degree at the Department of Advanced Control Systems Engineering, Saga University. His research interests include control of robot manipulators, real-time control, and force-free control.



Naruto Egashira received the B.S., M.S., and Ph.D. degrees all in electrical engineering from Saga University in 1984, 1986, and 1999, respectively. Since the year 2000, he has been an Associate Professor in the Department of Control and Information Systems Engineering, Kurume National College of Technology, Fukuoka, Japan. His research interests include mechatronic systems control and remote control.
Dr. Egashira is a member of the Society of Instrument and Control Engineers of Japan, the Robotics Society of Japan, and the Institute of Electrical Engineers of Japan.



Masatoshi Nakamura was born in Fukuoka, Japan, in 1943. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from Kyushu University, in 1967, 1969, and 1974, respectively. From 1973 to 1974, he was a Research Associate in Kyushu University. Since 1974, he has been with the Faculty of Science and Engineering, Saga University, where he is currently a Professor of Electrical and Electronic Engineering. His research interests include systems control theory and its applications, especially in the fields of power system control, thermal flow-control, robotics and biomedical engineering.
In addition to his involvement with the IEEE, Professor Nakamura is a fellow of the Society of Instrument and Control Engineers of Japan, the Institute of Electrical Engineers of Japan, the Robotics Society of Japan and the Institute of Systems, Control and Information of Japan.