Internet-based Teleoperation of a Robot Manipulator for Education

Xiaoli Yang[1], Qing Chen[2], Dorina C. Petriu[3], Emil M. Petriu[2]
[1]*Lakehead University, Thunder Bay, ON, Canada*
[2]*University of Ottawa, Ottawa, ON, Canada*
[3]*Carleton University, Ottawa, ON, Canada*
E-mail: lucy.yang@lakeheadu.ca ,
{qingchen, petriu}@site.uottawa.ca,
Dorina.Petriu@sce.carleton.ca

Abstract

In order to facilitate the study of the robot manipulator control for the users who cannot access the real robot manipulator, we are developing an Internet-based teleoperation system for a robot manipulator-Thermo CRS A465 which is located in the lab at Lakehead University. In this paper, the system architecture and software design will be introduced.

1. Introduction

Robot manipulators are being used in many fields such as industry, research and education. There are many existing teleoperation systems of robotics today. The first planetary mission - Mars Polar Lander (MPL) mission, uses Internet-based distributed ground operations where scientists and engineers collaborate in daily mission operations from multiple geographically distributed locations via the Internet [1][2]. The lander's robotic arm was controlled by a Web Interface for Telescience to get the information useful for planetary missions. Robot can also be employed in surgeries which are hazardous or far away from the human surgeons. A virtual reality robotic telesurgery simulation system is introduced in [3], in which the robot is controlled remotely to implement some surgeries, at the same time give the force feedbacks, videos back to the surgeon to get a better control. The mobile robots can be put in a museum as an interactive tour-guide robot which guides people through museum, explaining what they see along the way [4]. A virtual teaching system with a haptic interface was developed by [5] to facilitate the learning of Chinese characters and guide the movements. In [6], the mercury project was carried out to build a 6 DOF arm to allow users to dig and water the plants.

Due to the essential role of the robotics in different fields, it becomes important to learn how to control the robot manipulator. However, there are limited robot manipulator resources for a common user to access. In most situations, these robot manipulators are in the labs where only a few people can touch them. In order to facilitate the study of the robot manipulator control, especially for the users in Northwestern Ontario, we are developing an Internet-based teleoperation system of a robot manipulator. The robot manipulator - Thermo CRS A465 is located in the lab at Lakehead University. The user can implement the operations on the real robot manipulator at any place where the Internet access is available.

In this paper, we will give the architecture and the software design of the teleoperation system of our robot manipulator.

2. System Description and Architecture

There are two subsystems in this main system: one subsystem is the control of the 3D virtual robot manipulator and the virtual environment which simulate the real robot manipulator and its working environment. This subsystem can be used for off-line computers. Another subsystem is the real-time control of the real robot manipulator, which needs to be done on-line and connected to the robot manipulator in real-time.

Through the operations in this control system, the user can learn typical robot manipulator controls including how to solve different kinematics problems such as forward kinematics, inverse kinematics or velocity kinematics; how to implement the dynamics model; how to develop path planning through defining a reference trajectory by interpolating a sequence of points with which the robot manipulator can

0-7803-8817-8/04/$20.00 ©2004 IEEE.

understand the task; and how to implement linear control and nonlinear control for robot manipulators.

Through the user interface, the user can do simple controls without programming and get the basic knowledge about how the robot manipulator works. For example, if he wants the joints of the robot manipulator to rotate specified angles, he can define the orientation as the parameter for each joint in the function of the joint variables. Then the robot manipulator will do the corresponding movement according to the variable definitions. From this example, the user understands the principle of the forward kinematics problem and learns how to solve it. The user can also program on the virtual manipulator with C, MATLAB or other general purpose programming language to implement a series of complex control of the robot manipulator in this subsystem. One example is that the user can develop his own algorithm to get the joint variable values in terms of relevant position and orientation between the end-effecter and the specified goal. This algorithm can control the robot manipulator to move to the specified goal according to the calculated values. This example is a typical inverse kinematics control problem. Collision avoidance always works for all kinds of controls. Whenever the robot manipulator is detected to have collisions with objects in the virtual environment, its moving will be stopped automatically.

In the first subsystem, we simulate the robot manipulator and the environment with OpenGL. Fig. 1 shows the virtual robot control subsystem architecture. It includes the following components: the User Interface, Control models such as kinematics control model and dynamics model, the virtual robot manipulator and the environment, and the collision avoidance detection. Before the user has the solid knowledge and experience of how to control the robotic, he can practice the control on the virtual robot manipulator in the virtual environment, in order to protect the expensive robotics in the lab.

In the second subsystem, the user can control the real robot manipulator - Thermo CRS A465 in real-time. Fig. 2 shows the second subsystem's architecture. There are two main parts: the client part and server part. In the client part, the user can send control commands through the user interface to the real robot manipulator with the network connection. When the robot manipulator executes the commands, the real-time images acquired from the on-site camera will be transferred from the server side to the client side for the user to watch and compare with the virtual manipulation, at the same time, the virtual robot manipulator and virtual work environment can be updated in real-time following the changes of the real environment. On the server side, the robot manipulator is connected with the server computer through the controller. The controller will control the manipulator to implement the command.

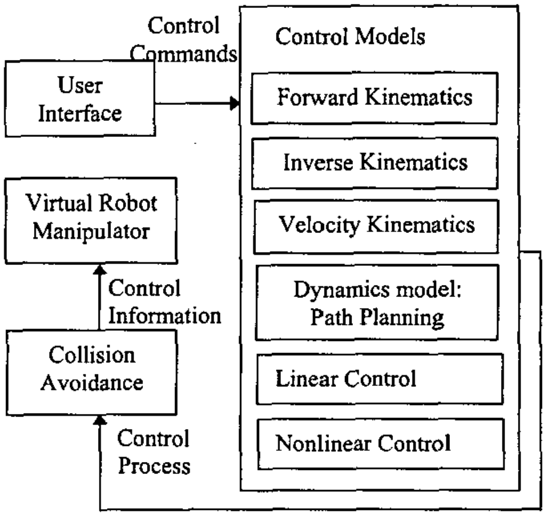

Fig 1. Virtual Robot Control Subsystem architecture

3. Software Architecture Design

Based on this control system design and architecture, we designed the software architecture.
Fig. 3 gives a sequence diagram that shows how the user does the inverse kinematics control of the robot manipulator in the first subsystem. The virtual robot manipulator can touch the point which the user specified through the Inverse Kinematics control. First, the user can choose the Inverse Kinematics control from the user interface. He needs to specify the goal point before the control starts. There is default inverse kinematics in this control system. If the user wants to practice the principle of inverse kinematics, he can create his own algorithm part. The control coordinator will trigger the default or user-defined inverse kinematics model after getting the command of inverse kinematics control and the goal points from the user. At the same time, the collision avoidance model will be run with the movement of the virtual robot manipulator. Both inverse kinematics and collision avoidance control work together on the virtual robot manipulator. The robot manipulator is controlled to move towards to the specified goal. When it is going to collide with the object in the virtual environment, it

will be stopped before the goal is gotten. A message window will pop out to describe the details of the result if the command is not executed completely.

Fig. 4 shows a sequence diagram which describes the sequence of inverse kinematics in the second subsystem. Besides the virtual robot manipulator control, the control is extended to the real robot manipulator in the remote site. As same as the control in the first subsystem, the user inputs the inverse

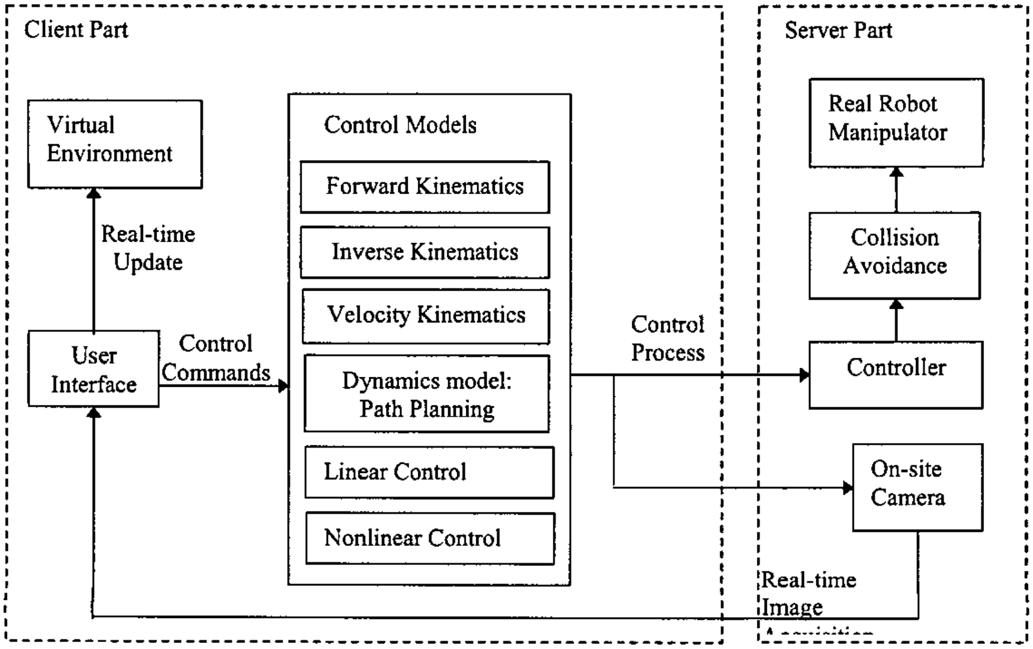

Fig 2. Real Robot Control Subsystem architecture

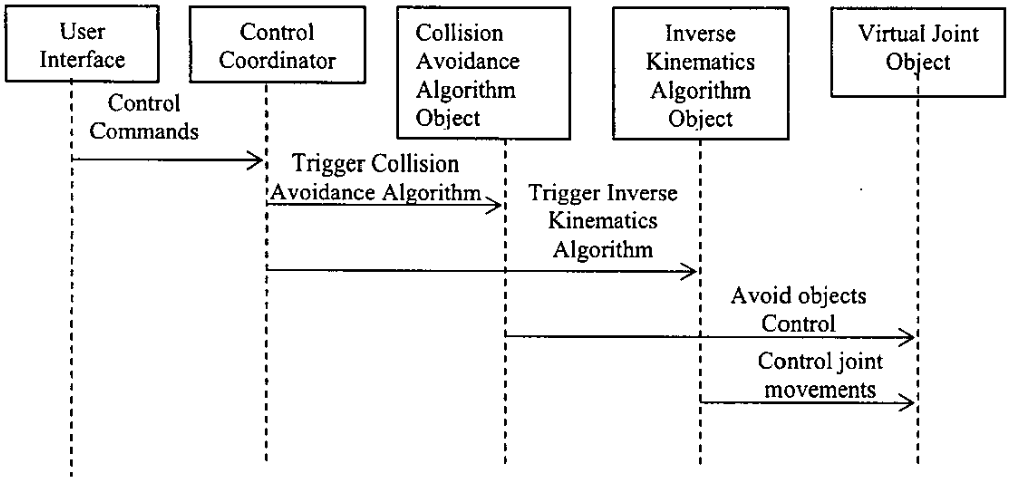
User Interface
Control Coordinator
Collision Avoidance Algorithm Object
Inverse Kinematics Algorithm Object
Virtual Joint Object
Control Commands
Trigger Collision Avoidance Algorithm
Trigger Inverse Kinematics Algorithm
Avoid objects Control
Control joint movements

Fig 3. UML sequence diagram illustrating the "Inverse Kinematics Control" of virtual robotics manipulator

kinematics command and the goal point through the user interface. This control information from the client side will be sent to the controller on the server side through TCP/IP Internet connection. The control coordinator triggers collision avoidance and inverse kinematics control objects. These two objects controls the real robot manipulator to implement the movement. The control command from the client side also triggers the image acquisiton object which gets real-time images from the camera on-site as the real robot manipulator moves. The real-time images will be sent to the image processing object on the client part to get the real-time environment updating information. The updating information is used to update the virtual robot manipulator and virtual environment. At the same time, the images can be shown on the client side for the comparation between the real environment and the virtual environment to get the better control.

The above description of these two sequence diagrams shows the examples of inverse kinematics control in the system. We are developing other controls for the user to learn and understand the principle of robotics.

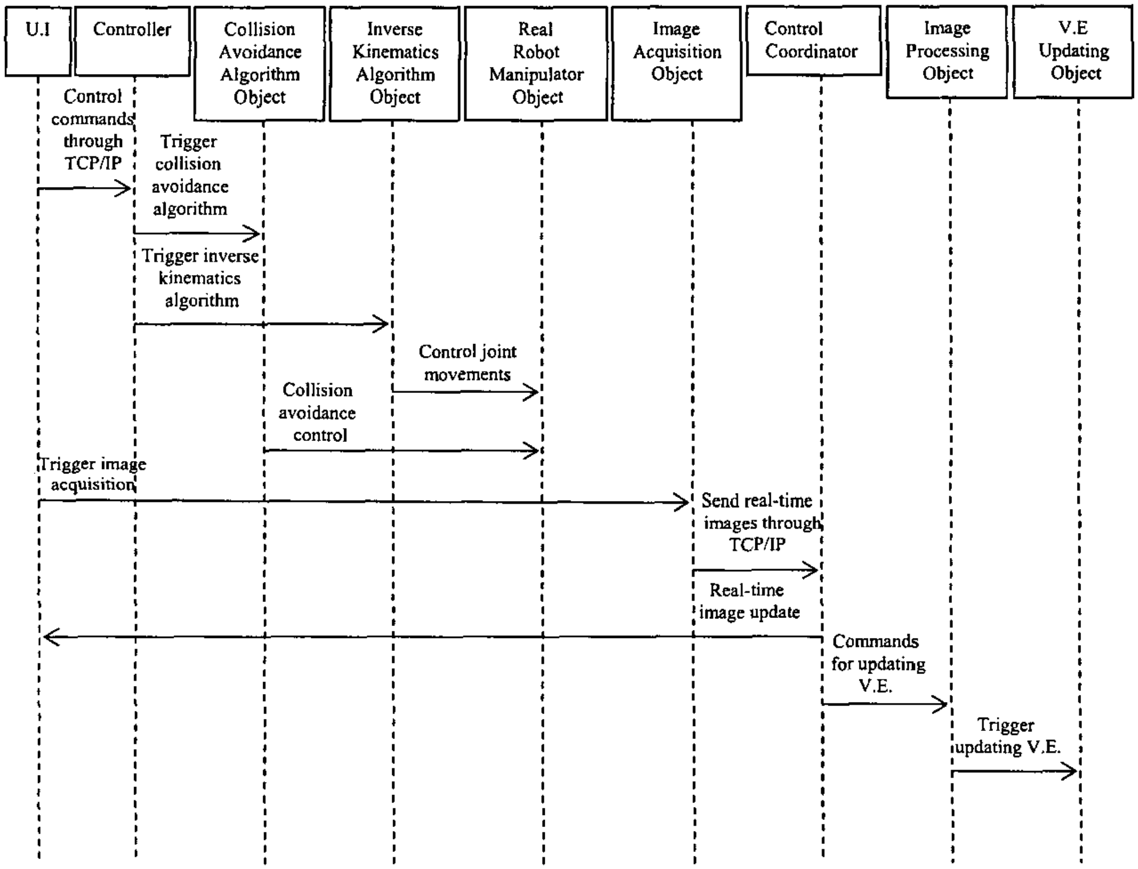
U.I
Controller
Collision Avoidance Algorithm Object
Inverse Kinematics Algorithm Object
Real Robot Manipulator Object
Image Acquisition Object
Control Coordinator
Image Processing Object
V.E Updating Object
Control commands through TCP/IP
Trigger collision avoidance algorithm
Trigger inverse kinematics algorithm
Control joint movements
Collision avoidance control
Trigger image acquisition
Send real-time images through TCP/IP
Real-time image update
Commands for updating V.E.
Trigger updating V.E.

Fig 4. UML sequence diagram illustrating the "Inverse Kinematics Control" of real robotics manipulator

4. Conclusions and Future Work

Learning how to control robot is an important field in the robotics study. In order to facilitate the learning process for the user who cannot access the real robotics, we develop a learning system which applies the 3D virtual environment techniques to the control of the robotic manipulator remotely.

Two subsystems are being developed in this research. The user can learn to control the virtual robot manipulators in the first subsystem. He can study topics such as forward and inverse kinematics, dynamic model, and linear or nonlinear control of the robotics. After he has enough confidence to operate on the real robot, he can use the second subsystem which will connect the real robot manipulator in the distant site. In the second subsystem, the virtual robot manipulator and virtual environment on the client side can be updated in real-time with the movements of real robot manipulator.

In the future, how to coordinate the controls on the robot manipulator from concurrent users should be solved. A more general control system can also be developed for different kinds of robot manipulators.

5. References

[1] Paul G. Backes, Kam S. Tso, Jeffrey S. Norris and Gregory K. Tharp, Internet-Based Operations for the Mars Polar Lander Missions. Proceedings of the 2000 IEEE International Conference Robotics & Automation, April 2000

[2] Paul G. Backes, Gregory K. Tharp, and Kam S. Tso, The web interface for telescience (WITS). In proceedings IEEE International Conference on Robotics and Automation, pages 411-417, Albuquerque, New Mexico, April 1997.

[3] Yoseph Bar-Cohen, Constantinos Mavroidis, Mourad Bouzit, Benjamin Dolgin, Deborah L. Harm, George E. Kopchok, Rodney White, Virtual reality robotic telesurgery simulations using MEMICA haptic system. Proceedings of SPIE's 8th Annual International Symposium on Smart Structures and Materials, 5-8 March, 2001, Newport, CA.

[4] Yoseph Bar-Cohen, Constantinos Mavroidis, Mourad Bouzit, Benjamin Dolgin, Deborah L. Harm, George E. Kopchok, Rodney White, Virtual reality robotic telesurgery simulations using MEMICA haptic system, Proceedings of SPIE's 8th Annual International Symposium on Smart Structures and Materials, 5-8 March, 2001, Newport, CA. Paper No. 4329-47.

[5] CL Teo, E. Burdet, and HP Lim, A Robotic Teacher of Chinese Handwriting, Proceedings of the 10th Symp. On Haptic Interfaces For Virtual Envir. & Teleloperator Systs. (HAPTICS'02).

[6] K. Goldberg, J. Santarromana, G. Bekey, S. Gentner, N. Rothenberg, C. Sutter, and J. Wiegley, Desktop tele-operation via the World Wide Web, Proc. IEEE Int. Conference on Robotics and Automation, pages 654-659, 1995.