An OS Platform Independent Architecture of Web-based Teleoperation for mobile robot

Deok-Hyeon Ko, Soon-Geul Lee

* Department of Mechnical Engineering, Kyunghee University, Yong-in, Korea
(Tel : +82-31-201-2945; E-mail: dhko@hotmail.com)

Abstract: The teleoperation system applies all of the industrial fields due to the development of the network infrastructure. It is one of the indispensable elements for controlling the robot at a remote sight and monitoring the limit or unknown environment. The common teleoperation robot system is what has the visual module to supply the network system and realistic UI to the existed robot system. Therefore, remarked that the fusion between modules and transmission of visual data the remarked the important element to improve the robot application in the various environments. Delay of development time by robot platform and noneffective communication among developers are also problem to approach.

In this paper we propose the independent teleoperation system. The main application language is JAVA in this system, which is applied JAVA API like JNI and JMF to construct the effective teleoperation system. The system has the both side communication system between sever and client as a basic structure. The visual data that is attached the robot at a remote sight is captured by JMF API and then is transmitted to the web browser called client by RTR protocol. JNI is used to connect between JAVA and the lower part application (sensor fusion, motion control.) of the robot programmed by various Native languages.

The proposed system is the application that can perform the elements, for instance transmission of visual data, the fusion of various native application modules and the effective network communication, with any platform.

Keywords: JAVA, JNI, JMF, Teleoperation

1. INTRODUCTION

Recently, the magnification of networks infrastructure and a development of internet technology cause the development of a teleoperation technology. The teleoperation approach using the web-browser as a common user interface has been progressed in many researches.[1~7] Telerobotics filed is divided into the telemanipulation for robot arm control and the vehicle teleopertiaon for vehicle robot control like a mobile robot. According to previous researches, the construction of teleoperation system consists of several parts.[1] ; (a) A socket module, which transmits and receives data to control robot at a remote area through network (b) A independent visual feedback module, which can transfer video between user and robot. (c) A management of various modules including socket, video, motion control, etc. to achieve unification, flexibility and extension

The robotics Hardware platform can be a personal computer (PC) format and various platforms according to the development situations. We will focus on implementation using PC based platform. Operation platforms for PC are divided into two groups; Windows and Linux. Researchers developed the solution for robot application with different development environment and programming languages depend on these two platforms. In different platform environments, research causes the delay of development period and the rise in cost of robot production.

Moreover, researchers are lack of communication for cooperation, so that it causes the delay in achieving the common purpose, the improvement of robot technology.

In this paper, we describe a teleoperation system for mobile robot which independence of OS (Operating System) platform. To construct platform independent application, we adapt JAVA programming language.

JAVA is useful for web-based teleoperation due to GUI like Applet being contained on a web-browser and flexible in application using network infrastructure. Although there are programming languages and development environments which become free on OS platform, JAVA is the most useful one than others as reasons about transplantation for Web documents. JAVA has some faults with the exception of advantage as language supported web.One is support interface hardware enough not like Native language. Other is capability how to support multimedia data. Communication data of high density as sound, visual and video etc., was needed for monitoring and observation at a sight area, so that JAVA should be flexible to process these datum to teleoperate a robot at a remote area.

To archive purposed system, we apply JNI (Java Native Interface) API to enhance capability of JAVA in terms of low level hardware control and JMF (Java Media Frame work) to rapidly transmit and safely process a visual data. That is, Java language can call for, transmit a variance to, and return it from the custom function consist of simple Native language operating hardware through JNI API. By adapting Classes of JMF, we can use various formatted multimedia data as like sound, video, and image. To realize a proposed architecture, we apply on a self-made Mobile Robot based on SBC (Single Board Computer) and general type of two wheel driving.

2. SYSTEM ARCHITECTURE

Fig. 1. System Configuration

The purposed system structure in Fig.1 is as follows.

The first part is the socket module to communicate between sever and clients. The sever application in a mobile robot and the client application embedded in a web-browser creates the sever socket and the client socket to exchange data each other. Each socket creates the multi-threads to secure independent data exchange channel for the accurate and efficiency communication.

The secondary part is the module for transmission and reception of visual data through network. A visual feedback module is necessary for a teleoperation technology. For search and navigation of a robot, this module should transmit the sensed visual information in unknown environment to user. The equipped visual system is the type of attached camera to guarantee the independence of mobile robot. It is interfaced by the main controller through USB port, and the system captures visual data with graphic library (etc., Windows: vfw , Linux : vfl) according to platforms. The visual data is transmitted to client with RTP(Real Time Protocol).

The final part is native programming language and interface module. Mobile robot can perform tasks in various environments due to its mobility and adaptability. There are two important parts for performing tasks. One is the sensor part to recognize and perceive the information of unknown environments. The other is the motion part to move rapidly and accurately. They are programmed by the native programming language familiar with hardware to interface with main controller because these parts are organized by the hardware which is different from PC as a main controller. Various external modules should be packaged like class for integration and flexibility with a main controller as these modules are produced by cooperated development. The main controller should be efficiently communicated with sub-function module and be in charge of total solution to manage Sensor Data integration and Motion planning.

To realize these functions, we construct the system by using JAVA language which support the multi platform and apply various API

2.1 JNI(Java Native Interface)

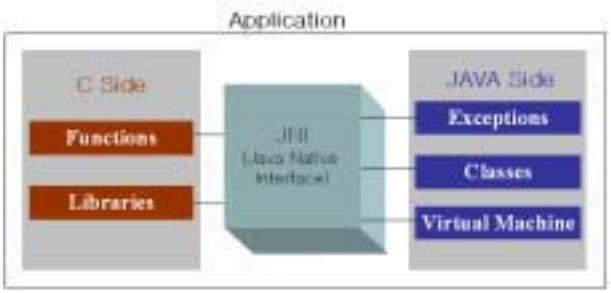

Fig 2. JNI Structure

The Java Native Interface (JNI) is the native programming interface for Java that is part of the JDK (Fig.2). By writing programs using the JNI, you ensure that your code is completely portable across all platforms. The JNI allows Java code that runs within a Java Virtual Machine (VM) to operate with applications and libraries written in other languages, such as C, C++, and assembly. In addition, the Invocation API allows you to embed the Java Virtual Machine into your native applications.[8]

3.2 JMF(Java Media Framework)

JMF provides a platform-neutral framework for displaying time-based media. The Java Media Player APIs are designed to support most standard media content types, including MPEG-1, MPEG-2, QuickTime, AVI, WAV, AU, and MIDI. Using JMF, you can synchronize and present time-based media from diverse sources. Existing media players for desktop computers are heavily dependent on native code for computationally intensive tasks like decompression and rendering. The JMF API provides an abstraction that hides these implementation details from the developer. For example, a particular JMF Player implementation might choose to leverage an operating system's capabilities by using native methods. However, by coding to the JMF API, the application or applet developer doesn't need to know whether or not that implementation uses native methods. [9]

3.2 RTP(Real Time Protocol)

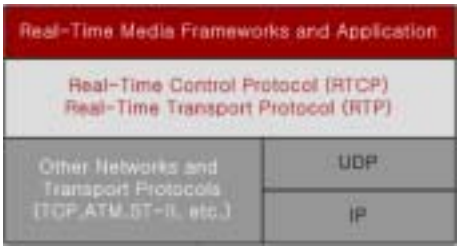

Fig. 3 RTP Architecture

To send or receive a live media broadcast or conduct a video conference over the Internet or Intranet, you need to be able to receive and transmit media streams in real-time.

The JMF APIs that support RTP are found in the javax.media.rtp,javax.media.rtp.event,andjavax.media.rtp.rtcp packages. RTP provides end-to-end network delivery services for the transmission of real-time data. RTP is network and transport-protocol independent, though it is often used over UDP. RTP can be used over both unicast and multicast network services. Over a unicast network service, separate copies of the data are sent from the source to each destination. Over a multicast network service, the data is sent from the source only once and the network is responsible for transmitting the data to multiple locations. Multicasting is more efficient for many multimedia applications, such as video conferences. The standard Internet Protocol (IP) supports multicasting.

Table 2. Mobile Robot Specification

Parts	Specificatiuon
Main Contoller	SBC (Single Board Computer) Supported PC104 interface. Pentium Ⅲ 800, 256 M.
Motion	75W dc motor with encoder. LM629 motion Controller (CAN interface with SBC)
Sensor	Vision System with Common USB Camera.
Communication	802.11b Wireless Network (PCMCIA Card).

4. SERVER SYSTEM

Robot as Sever system is organized with the motion part for a robot mobility, the communication part for linkage of others and the sensor part for an environment awareness and monitoring.

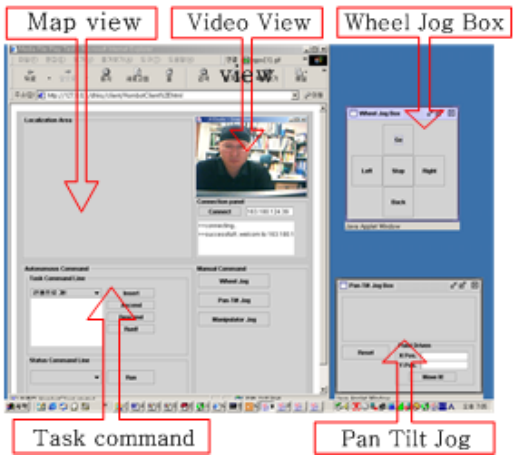

Fig. 3 Client GUI based on Web-Browser

5. CLIENT SYSTEM

Client part has the capacity which transmits command data to control a robot at a remote area and which shows user a visual data from sever. Data of both client capacities are transmitted through single IP address on this system. Applet can't connect to several sever by self security system so that it use single IP address to connect. When the connection with sever is complete. Sever is recognized client IP address by connection, makes a command-transmission and a visual transmission port and communicate with client.

5. ALGORITHM

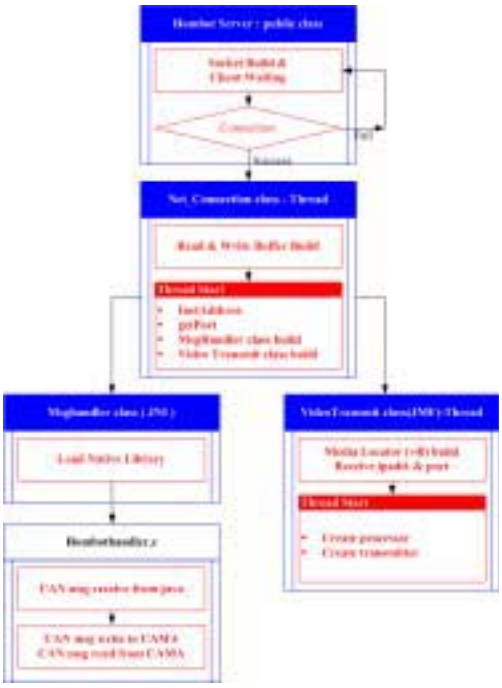

Fig. 4 Server Algorithm

5.1 Server Algorithm

A sever will be operated by client's command and control constructed modules of a robot. The proposed system is designed on Windows and Linux as a common OS. The performance algorithms by sequence of command in order are as Fig 3. When sever start to operate, the sever application reset to check each consist part. After resetting, it makes a sever socket and wait for a connection of client. A sever recognizes a connection demand from a client and accept it. Then it creates input/output-buffer to communicate with client. After creating buffers, a sever recognizes a client IP with InetAddress method and creates two classes simultaneously. One is MsgHandler class. It uses JNI for a robot performance according to communication with client. That is, it controls a robot through interface a hardware driver which are programmed by Native language according to message from client. The other one is VideoTransmit class. Video transmission class transmits a visual data being captured by USB camera as a sensor of sight. It progress the steps, which are capture, synchronize, transmission, with JMF API as a JAVA multimedia API. The transmission of a visual data is applied thread as a independent class from message communication class in order to keep monitoring environments surrounded a robot

5.2 Client Algorithm

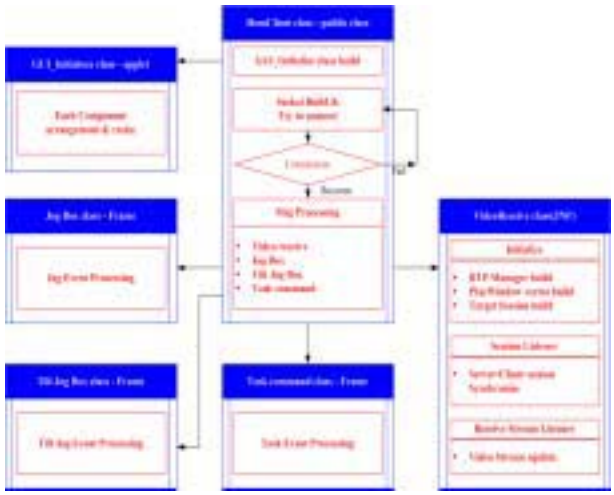

Fig. 5 Client Algorithm

The client is constructed by java applet form on a web-browser. The teleoperation control panel as an applet has an advantage to control a target robot with the terminal unit which is not to be tied to area. In addition, it supplies a visual and familiar GUI with SWING applet model. The client is started by performing web-browser. To transmit various messages effectively, robot tasks is separated into some group. The class for receiving visual data is programmed independently as a thread to communicate safely. Client progresses, connecting a sever, a command transmission and receiving a visual data, are described in Fig 4. Client is stared to be operated by running web-browser. A user connects a server through a unique IP of a mobile robot, and then a mobile robot control board is loaded as Fig 2. We divided class into some groups depend on robot tasks to achieve effectively in various messages. The class for receiving a visual data is made with thread to safely communicate with sever.

5. RESULT

In this paper, we proposed an independent solution of platform for the efficient teleoperation system. This system simplifies the complex teleoperation solution to simultaneously communicate a visual data on the teleoperation system by using the network for communicating commands. Moreover, we proposed the extensibility ability to fuse the modules operating hardware device based on a native language.

In the future, we will evaluate the performance of the proposed system, two types of methods, the comparison of visual date transmission rates and the measurement of average transmission/return commands time by Round Trip Time method, are performed on the each platform. The comparison which is according to the network level is performed in order to compare to the rate of efficiency on the network. We hope that the estimation of mobile robot position researches in the dynamic environment will progress to approach to the irregular network time delay, one of the negative things of teleoperation system.

REFERENCES

[1] Goldberg, K., et. al., “Desktop Teleoperation via the World Wide Web”, IEEE Conference on Robotics and Automation, Nagoya, Japan, May 1995.

[2] Sebastien Grange, Terrence Fong and Charles Baur, “Effective Vehicle Teleoperation on the World Wide Web”, IEEE International Conference on Robotics and Automation, San Francisco, CA, April 2000.

[3] Jussi Suomela, “Tele-presence aided teleoperation of semi-autonomous work vehicles”, Helsinki University of Technology (AS-84.147 Kurssimateriaali)

[4] C. Röhrig, A. Jochheim, “Java-based Framework for Remote Access to Laboratory Experiments”, IFAC/IEEE Symposium on Advances in Control Education, ACE 2000, Gold Coast, Australia, December 2000.

[5] D. Song and D.B. Kaber ,”Web-based interface design for teleoperation”, In the Proceedings of the XIVth Triennial Congress of the International Ergonomics Association and 44th Annual Meeting of the Human Factors and Ergonomics Society, Santa Monica, CA,2000, pp 449-452.

[6] Huosheng Hu., et. al “Internet-Based Robotic Systems for Teleoperation”, International Journal of Assembly Automation, 21:2 (2001) , pp 143-151.

[7] A. Khamis, D.M. Rivero, F. Rodriguez, M. Salichs, "Pattern-based Architecture for Building Mobile Robotics Remote Laboratories." Proceedings of IEEE International Conference on Robotics and Automation, 2003 (ICRA'03), May 12 to May 17, 2003, Taiwan.

[8] Sun Micro systems, “Java Media Framework API”, http://java.sun.com/products/java-media/jmf/index.jsp.

[9] Sun Micro systems, “Java Native Interface”, http://java.sun.com/docs/books/tutorial/native1.1.