ICAT '99

Development of R-Cubed Manipulation Language
-Implementation and Evaluation to a RCML System-

Wei-Chung Teng[1], Dairoku Sekiguchi[1], Akira Nukuzuma[2], Naoki Kawakami[1], Yasuyuki Yanagida[1], and Susumu Tachi[1]

[1]Tachi Laboratory,
School of Engineering, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656 JAPAN
{waldo, dairoku, kawakami, yanagida, tachi}@star.t.u-tokyo.ac.jp
[2]Takatsuki Laboratory, Minolta Co. Ltd.
1-2,Sakura-Machi,takatsuki-Shi, Osaka, 569-8503 JAPAN
nuu@eie.minolta.co.jp

Abstract

R-Cubed (R^3: Real-time Remote Robotics) is the concept and technology which attempts to provide a solution for people to telexist anywhere in the world by controlling remote robots as his avatars in real time base through the network. An R-Cubed system is supposed to allow people to control remote robots from terminals installed on homes, offices, or any public booths connected to Internet or other dedicated networks. As part of R-Cubed research, we proposed RCML/RCTP structure as a solution on implementing R-Cubed systems. To examine and verify practicability of RCML/RCTP structure, we also implemented an RCML system based on RCML and RCTP specification on a low-end platform. In this paper we will show how we implement the experimental system following the specification of RCML and RCTP. Observation and evaluation to the system results in some hints on designing next version of specification and are discussed in detail.

Key words: Telexistence, R-Cubed, RCML, RCTP, VRML

1. Introduction

R^3 stands for Real-time Remote Robotics [1], is a long-range project originated by Japanese Ministry of International Trade and Industry (MITI) in 1995. The concept of R^3 is to construct an infrastructure that everyone in the society can freely telexist through a network. In R^3 society there would have R-Cubed booths installed in home, offices, or public places quipped with teleoperation systems which people can use to control remote robots as if the robots be his/her avatars. Some other words such as networked telexistence or networked robotics are also used to express the concept of R^3 [2]. Figure 1 shows the concept of R^3 system. In the client-server systems each robot site includes its local server system. The robot type varies from a humanoid (high end) to a movable camera (low end). A virtual robot can also be a controlling subject, so we can telexist in virtual worlds as in remote environments.

On the other side, each client has a teleoperation system ranging from a control cockpit with master manipulators and a HMD (high end) to an ordinary PC equipped with mouse and maybe a joystick (low end). In the meantime real-time data transfer channel is employed to provide a basis for users and robots to communicate with each other. That is because only with real-time response can human retains the sensation of presence.

Towards the realization of R^3 concept, we proposed R-Cubed Manipulation Language (RCML) and R-Cubed

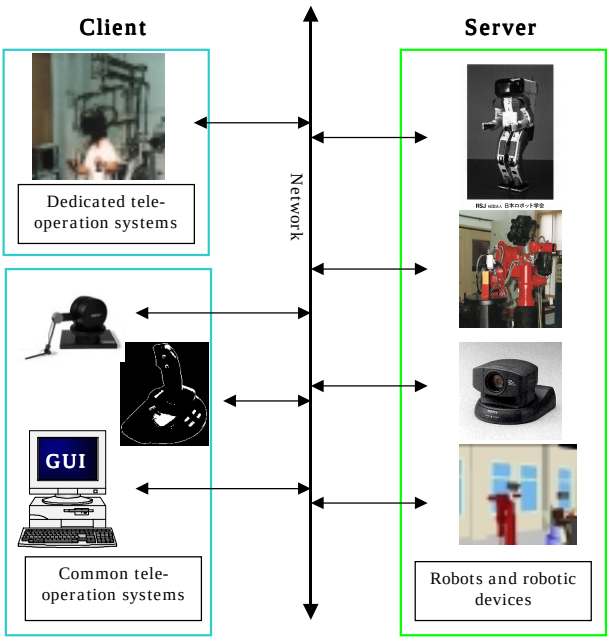

Fig. 1 An R^3 system should supports large scale of teleoperation systems and robotic devices

Transfer Protocol (RCTP) as a solution on constructing R^3 systems [3]. The RCML/RCTP approach is designed from the end-user's standpoint, keeping "how the users access the remote *real* environments" in mind. And on the early stage our study focuses on implementations on the low-end systems.

2. Related Works

Although networked telexistence, or networked robotics, is still a new research field, more and more successful cases are reported [4-9]. However, some researches aim on particular purposes such as space work [6] or medical treatment with dedicated systems that considered not intuitive to apply on general application. On the other hand, WWW browser seems to be a possible candidate for general-purpose networked robotics operation platform, since many approaches apply WWW browser to operate remote robots executing simple operations such as browsing inside building [8][9] or painting [10]. However, the innate characteristics of WWW Browser's structure implies problems on constructing a controlling platform and results the whole system in constrained and not intuitive structure [3].

On the case of virtual environment, since the VRML 1.0 standard was specified in 1994, the use of VRML has become the most popular way to construct and access virtual worlds. VRML 2.0 provides the mechanism on designing and handling the actions of objects, and has become an ISO standard known as VRML97 [11]. By the way, although some implementations show the possibility on controlling virtual robot modeled with VRML [12], a standard way to model controllable mechanism and sensory information is still necessary for various types of robots.

2. Notes on RCML and RCTP

Before we discuss the experimental implementation, a brief introduction on the concept and characteristic of the RCML/RCTP system is showed below.

2.1 Client-Server Architecture

The RCML/RCTP system is of client-server architecture as R^3 system. Server stands for the controlling system software running at the robot site, and client stands for the software running at the user's computer.

2.2 Objects

The concept of object is introduced o define the functional input/output units on the system. According to their functionality, the objects are grouped into three categories: *system object*, *input object*, and *output object*.

System objects include server object and client object, which refer to, in respective, the server and client's computer hardware and the RCML/RCTP software running above it, in the base that only one client can control the robot at a time.

Input/output objects can still be classified into five types according to the type of data they handle: *video input/output object*, *audio input/output object*, *control input/output object*, *text input/output object*, and *binary input/output object*. Each type of output object is the counterpart of corresponding type of input objects.

2.3 Translators

RCML/RCTP system is designed to be applicable for various types of robots and devices, and the way data transferred over the network should be standardized and be independent of the robots and devices connected at each site. To satisfy this requirement, software modules called *translators* are provided to translate the specific-format data into standard format and vice versa. For example, the data returned from some 6 DOF position sensor may contain 6 degrees for 6 joint respectively. The position data is converted by input translator into standard format, say coordinate in Cartesian coordinate system, before it's transferred to the other site. The transferred position data would be converted by output translator of the opposite site again to suite the specific output device.

An I/O device, when coupled with its translator, is treated as an input/output object in RCML/RCTP system. Figure 2 shows the relationship between object, translator, and device.

2.4 Structure of RCML

The R-Cubed Manipulation Language is designed as a file format for describing real world or virtual spaces and for robot characteristics such as degrees of freedom, control parameters, and sensor information.

Since VRML provides a standard and popular way to construct interactive three-dimensional objects and virtual worlds, we adopt VRML97 as the base format to

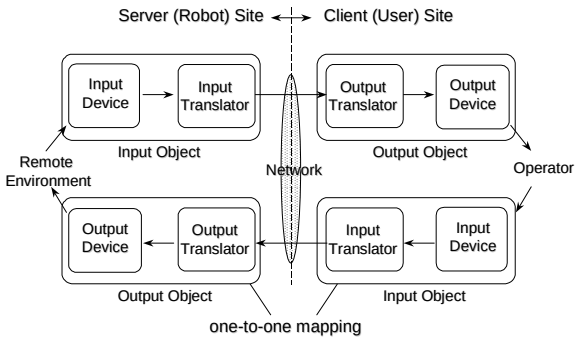

Fig.2 Functional units and information flow in the RCML/RCTP system

describe remote environments, controlled robots themselves, and robot characteristics. A typical RCML file should include two parts: a virtual world, which is the "copy" of real remote world, written as normal VRML files, and a RCML-defined PROTO node containing control parameters and sensory information description.

The structure of RCML extension nodes is constructed in the same way with *objects*. It contains 3 levels with the RCML_Robot grouping node as the "root" node. System object nodes and Input / Output object nodes are contained in RCML_Robot and stand for system objects and Input / Output objects respectively. Control objects are classified furthermore into small unit RCML_ControlInputData or RCML_ControlOutput Data nodes. Table 1 lists all kinds of nodes and shows their relationship.

2.5 Phases on RCTP

RCTP is designed to cooperate with RCML. The two main jobs for RCTP are to negotiate for assigning effective pairs of input/output devices in robot and user sites, and to transfer the control commands from user and status information from robot in real-time.

The process of a RCTP connection is divided into 3 phases. In the beginning the users access some Web pages describing the information of controlling robots, and then download the RCML file. RCML browser is invoked at this time to parse the RCML file. This part is called *the greeting phase*.

After the user picks out the controlling objects and decides the corresponding input device, RCML browser would try to build a network connection with RCML server. After connection is established, RCML browser would request to get the control permission of robot's objects. If the requested device is free and functions well, server will assign a unique ID number to the device and acknowledge it back with all initial characteristic values appended. All this process should follow the specification of HTTP/1.1 [13], and the part is called *the negotiation phase*. If negotiation phase completes smoothly, RCML browser would send the GO method in order to start the controlling phase. Control messages and system information transferred in this phase are binary based and contrived to meet the real-time needs, however the Video/Audio and other Sensory information are left to dedicated protocols. This phase is call as Live Session Phase.

3. Experimental RCML System

To examine and verify practicability of RCML/RCTP structure, we implemented an RCML system based on RCML and RCTP specification on a low-end platform. The structure of the system is shown in Figure 3. On the server site a direction controllable video camera and a notebook PC are fixed on the movable robot, where a

Table 1. List of Nodes in RCML

Level 1 Node		RCML_Robot
Level 2 Node	System Object Node	RCML_Server RCML_Client
	Input Object Node	RCML_VideoInput RCML_AudioInput RCML_ControlInput RCML_TextInput RCML_BinaryInput
	Output Object Node	RCML_VideoOutput RCML_AudioOutput RCML_ControlOutput RCML_TextOutput RCML_BinaryOutput
Level 3 Node	RCML_ Control Input Node	RCML_ControlInputData
	RCML_ Control Output Node	RCML_ControlOutputData

PCMCIA wireless LAN card is used to connect the notebook to the network. For most efficiency we use C language to code server program. The server program controls rotation and movement of the robot, and direction of video camera.

On the client site, the whole client application is executed on a low-end PC (Pentium MMX 200MHz with 64MB memory) with Joystick and ADL-1 equipped. ADL-1 is a 6DOF position sensor designed originally for measuring position and direction of human head, however we use it as a three-dimensional position input device. For higher portability, the main body of client program is written in Java language, with drivers of physical devices written in C language. The VRML browser (Sony Community Place Browser) is used to handle VRML part of the RCML file, and video conferencing program (CU-SeeMe) is used for video stream transfer. These helper programs greatly shorten the development period. How the helper programs cooperate with RCML client program is showed on Figure 5. Because RCML is defined as an extension of VRML, all Java program is linked from VRML file by Script nodes, by this way we could update VRML world

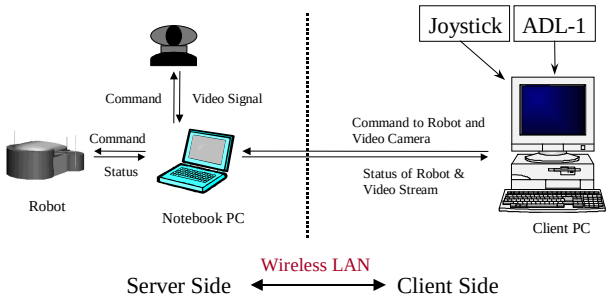

Fig.3 Hardware structure of the RCML system

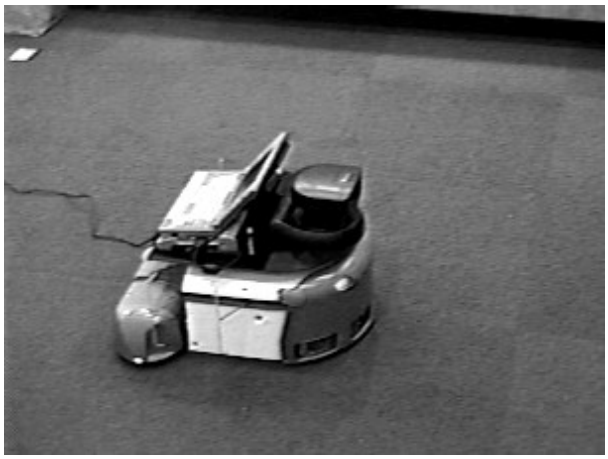

Fig. 4 Photo of the controlling robot

synchronized with remote environment.

In this experimental system, RCTP is implemented based

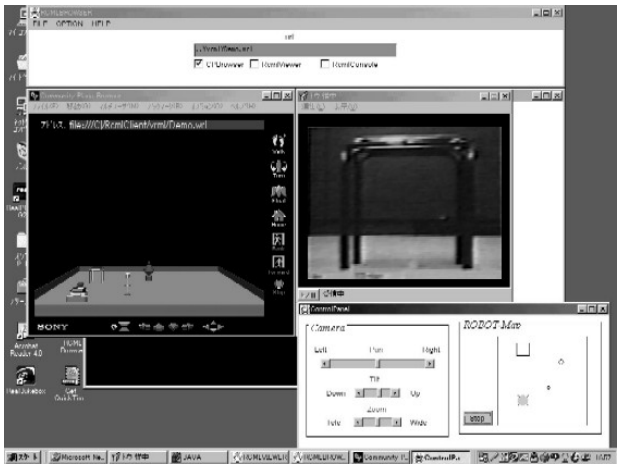

Fig. 5 Screen of client program with scroll bar and 2D map input objects

on TCP/IP protocol. On the beginning RCML file and translators are already in client site's computer, so the greeting phase is omitted. However, negotiation phase and live-session phase is faithfully realized. Users are asked to pick up his/her favorite input device if the request for controlling robot is permitted. Some common GUI, joystick, and ADL-1 are supported in this system, and they are treated in the same way by RCTP clone. Table 2 lists all objects in this RCML system. I/O objects of the same type can be paired up as defined in the RCML specification.

4. Results

Comparing to approaches using CGI and HTTP, our system provides flexible and compact structure to handle continuous command stream that is necessary for R^3 system. However, due to narrow bandwidth (2Mbps) of wireless LAN in the experimental system, time delay becomes obvious when high quality video stream is chose. Next version of RCTP should has suitable

Table 2. Objects in the RCML system

	Server Site	Client Site
System Object	Server Program	Client Program
Video Input Object	Video Camera	
Video Output Object		Video conferenc-ing program
Control Input Object		Common GUI (slider bar, text field, button, etc.), Joystick, ADL-1
Control Output Object	Direction and Zooming of Camera, Movement and Direction of Robot	

mechanism to negotiate how controlling command and sensory information share the limited bandwidth. During development period, we also found that VRML grammar and its Java platform class's definition did restrict future extension of RCML. Approach such as XML based RCML is under consideration.

4. Conclusion

In this paper, some implementation related topics on RCML and RCTP are picked up and compared to the experimental system. Practicability of RCML/RCTP structure is verified as this system treat all kind of input and output objects the same way by standardize the data type and format of object. However, further study on characteristic of user interface is necessary for constructing general-purpose teleoperation system.

References

1. MITI of Japan, R-Cubed WG ed.: "R-Cubed", *Nikkan Kogyo Shimbun* (1996).

2. S. Tachi: "Real-time Remote Robotics – Toward Networked Telexistence," *IEEE Computer Graphics and Applications*, pp. 6-9 (1998).

3. W. C. Teng, A. Nukuzuma, N. Kawakami, Y. Yanagida, and S. Tachi: "Development of R-Cubed Manipulation Language - The Specification of RCML and RCTP –," *Proc. of the 8th International Conference on Artificial Reality and Tele-existence*, pp.156-162 (1998).

4. R. Simmons: "Xavier: An Autonomous Mobile Robot on The Web," *Preprints of IROS'98 Workshop 'Robots on the Web'*, pp.43-47 (1998).

5. P. Saucy and F. Mondada: "KhepOnTheWeb: One Year of Access to a Mobile Robot on the Internet," *Preprints of IROS'98 Workshop 'Robots on the Web'*, pp.23-29 (1998).

6. Y.Wakita, S. Hirai, K. Machida, K. Ogimoto, T.Itoko, P. Backes, and S. Peters: “Application of Intelligent Monitoring for Super Long Distance Teleoperation,” *Proc. Of IROS’96*, pp.1031-1037 (1996).

7. A. Kheddar, C. Tzafestas, P. Coiffet, T. Kotoku, S. Kawabata, K. Iwamoto, K. Tanie, I. Mazon, C.Laugier, and R. Chellai: “Parallel Multi-Robots Long Distance Teleoperation,” *Proc. of ICAR’97*, pp.1007-1012 (1997).

8. S. B. Goldberg, et al.: “DIGIMUSE: A interactive telerobotic system for remote viewing of three-dimensional art objects,” *Preprints of IROS’98 Workshop ‘Robots on the Web’*, pp.55-59 (1998).

9. S. Maeyama, S. Yuta, A. Harada: “Mobile Robot in the Remote Museum for Modeling the Evaluation Structure of KANSEI,” *Proc. of 7th IEEE International Workshop on Robot and Human Communication, Vol. 1*, pp.315-320 (1998).

10. M. R. Stein: “Painting on the World Wide Web: The PumaPaint Project,” *Preprints of IROS’98 Workshop ‘Robots on the Web’*, pp.37-42 (1998).

11. http://www.vrml.org/technicalinfo/specifications/vrml97/index.htm: “VRML97 Specification, ISO-IEC 14772-11997,” *Web3D Consortium* (1997).

12. http://www.robotic.dlr.de/STUDENTS/Martin.Rohrmeier/robot/robot.html

13. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Berners-Lee, T.: “Hypertext Transfer Protocol -- HTTP/1.1,” *RFC 2068, UC Irvine, Digital Equipment Corporation, M.I.T.* (1997).