Demonstration of the Low-Cost Virtual Collaborative Environment (VCE)

David Bowers, Leticia Montes, Angel Ramos, Brendan Joyce, Dr. Ron Lumia
NASA Center for Autonomous Control Engineering (ACE)
The University of New Mexico

Abstract

This paper demonstrates the feasibility of a low-cost approach of remotely controlling equipment. Our demonstration system consists of a PC, the PUMA 560 robot with Barrett hand, and commercially available controller and teleconferencing software. The system provides a graphical user interface which allows a user to program equipment tasks and preview motions i.e., simulate the results. **Once** satisfied that the actions are both safe and accomplish the task, the remote user sends the data over the Internet to the local site for execution on the real equipment. A video link provides visual feedback to the remote sight. This technology lends itself readily to NASA's upcoming Mars expeditions by providing remote simulation and control of equipment.

Keywords: virtual collaborative environment, control, robotics, remote access, distance learning, NASA Mars expedition.

Introduction & Project Overview

The aim of this project is to develop and demonstrate the applicability of remotely controlling equipment in a cost effective manner. A remote user can program and simulate t he actions the equipment is to perform. When satisfied with their program the data are sent to the local sight for actual execution. Teleconferencing software and a camera at the local sight provide real-time viewing of the workspace to the remote user. This is demonstrated through the use of the PC, the PUMA 560 robot[1] with Barrett hand, and commercially available control ler[2] and teleconferencing [3] software.

For our application we have remotely programmed the PUMA 560 with Barrett hand to go to an object, pick the object up and move it. In this paper we will discuss the components of the low-cost VCE system, use of the system, and some applications.

Components of the Low-Cost VCE

The design of the low-cost VCE uses a PC platform. The software is broken into two components: the equipment controller software and the real-time viewing (teleconferencing) software. The equipment being controlled is the PUMA 560 robot with Barrett hand.

In this project we use the Cimetrix Open Architecture Controller. This consists of an amplifier/control module and Cimetrix controller software. The software is broken into three different components, one for program development, one for simulation and one for execution.

There were several elements to consider in determining what the data transmission and communication scheme would be. We want to provide access to University labs (the "local" site) and their equipment (robots), through an interactive environment controlled by LAN/WAN accessible computers, from a remote site computer. The local site must receive file transfers and transmit live video data. The remote site must transmit files and receive live video. A previous phase of this project[4] successfully used a remote site composed of a terminal with direct WAN access to the local site for the file transfers, and a stand alone PC with a direct phone link

(POTS) to another stand alone PC at the local site to provide a channel for video. The current phase has some unique requirements creating the need for a new solution. The first constraint is that the remote platform is a laptop running Windows NT 3.51 (necessitated by the latest Cimetrix software). Although there are several video conferencing systems available, it was difficult to find hardware and software that could run on Windows NT. Next, the only available communication channel is POTS, New Mexico does not have Integrated Services Digital Network (ISDN) capabilities. Lastly, the local LAN we are accessing does not have dial-up access.

This combination of constraints on the remote and local systems proved to be a major obstacle to finding an off-the-shelf solution. A more universal solution consists of some type of WWW accessible web site, linked to the local site, through which a remote site running a web browser would access the video feed. Due to the complex, custom programming inherent in such a configuration, we opted for a more task specific solution described below.

The teleconferencing software chosen, *Enhanced CU-SeeMe,* is made by White Pine. This software provides person to person, group conferencing, and large audience broadcasting with video, audio, chat window and white board communications over the Internet or any TCP/IP network. The camera used is called a Connectix QuickCam, which transmits video in black and white. This particular hardware was chosen because it can run on Windows NT and is compatible with *Enhanced CU-SeeMe*. The teleconferencing software and the Connectix QuickCam were relatively low in price and met the needs of the VCE system.

The resultant configuration required the addition of a LAN linked PC to the local site. The local PC, running Windows NT 3.51, is equipped with a modem (for dial in access), and a gray scale camera (QuickCam), by Connectix for the video feed. The remote site would dial into the PC, then launch a viewer application (provided by Connectix) to begin seeing the live video.

One concern is secure and safe transport of data. VCE would incorporate the use of a firewall, which requires a password and separate program for admittance. The program and password would then be electronically mailed to the remote user. Any other party trying to interrupt the transport of data will require these components. Use of VCE also requires the remote user to log into the local users system and transfer data. Another way of accomplishing this is to mail the program electronically, so that the interchanging of passwords does not occur. It must be understood by both remote and local user that robotic simulations are written in Cumulation. The *Enhanced CU-SeeMe* also incorporates security into the software, such as password usage, caller identification, and other conference and inbound call security.

The equipment being controlled is the PUMA 560 robot. This is a robotic arm containing six degrees of freedom, "Each link of the robotic arm is connected to another link at a joint, and through each joint passes one or more axes around which the links of the arm rotate. At the end of the arm is a Barrett hand used for gripping objects. The links are driven by a permanent magnet DC servomotors driving through its associated gear train. The position and velocity of each joint are needed to control the PUMA 560, and are programmed using CIMBuilder. The CIMControl software is used for real-time control of the PUMA 560. CIMControl outputs control signals to a servo board, which connects various power amplifiers to control robot motion.

Task Development

Task development is implemented via the Cimetrix Open Architecture Controller using CIMBuilder. The application is then run in Cumulation, a graphic simulation package, to preview the application. Once everything is operating correctly CIMControl is used to control the actual mechanism. Our task development goal was to use the system to remotely program the

PUMA 560 robot to pickup an object and move it following the sequence of programming, simulation, and finally execution. This proceeded as follows:

1) Build a model of the robot and the work cell

The first step in programming equipment is to make a model of the equipment and the work space in which it will operate. This is accomplished by setting nodes, geometric dimensions, and degrees of motion. The PUMA 560 contains 6 degrees of freedom and is fitted with a Barrett hand for gripping. A custom **work** cell that includes a tool table and a peg was built with CIMBuilder for use with Cumulation.

2) Program the robot

With the work cell and equipment defined the user programs the equipment to do a desired set of tasks within the work cell. This maybe accomplished in a variety of ways. One way is using CIMBuilder, a general purpose application development tool consisting of a Graphical 'User Interface (GUI) builder and a program builder allowing the user to attach a GUI to the program. Another way is to write a C program using the applications interface (API) functions from the Cimetrix Open Development Environment (CODE) Library. Here again the user programs the equipment to do a desired set of tasks. In both cases the program is built using CODE API libraries. For example to cause the robot to move to a peg, pick it up, put it down and return to its original position the user would go through a sequence similar to this:

MoveNearNode - user specifies target, tool, and distance
MoveToNode - user specifies which node or object
MoveSingleAxis - close the gripper on the peg, specify axis and value
AttachNode - attach peg to the robot gripper
MoveNearNode - move back to starting position
MoveToNode - put the peg somewhere else
MoveSingleAxis - open the gripper
AttachNode - put the peg down
MoveNearNode - return the robot to some known position

Each of these functions is defined in the CODE API libraries. All these functions may be combined **into** a procedure if desired. This procedure can then be run as a stand alone program. This procedure performs is a basic pick and place operation.

3) Simulate the program

Once either of the two programming approaches from step 2) has been done, the movement is simulated using Cumulation. This gives a simulation of the equipment's movement in the work space, allowing the user to test the program without turning on the robot. This type of simulation is quick and safe for the user. Cimetrix displays a window with a complete 3D-model of the robot and work cell. The motion and picture on the screen simulate the real robot. After the simulation is done, the user will decide on modifications or improvements if required. If no modifications are required the complete program is ready to be sent to the local site for test on the real robot.

4) Execute the program

When satisfied with the program and the results, CIMControl is **used** to move the actual equipment. Figures 1 and 2 show the task development cycle. The same program can be run with a completely different robot so long as the new robot can reach the designated object, i.e. the Peg. The simulation and control programs, Cumulation and CIMControl, are interchangeable. The application can not determine whether it is connected to the simulation or the real time robot controller,

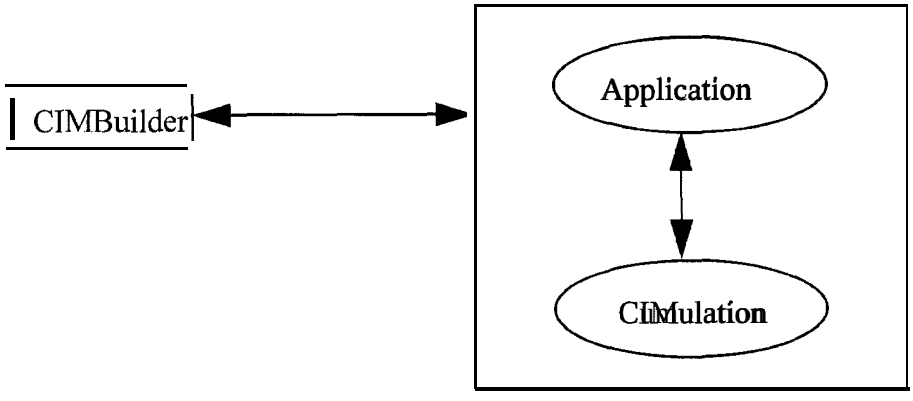

Figure 1. Using CIMBuilder to develop an application

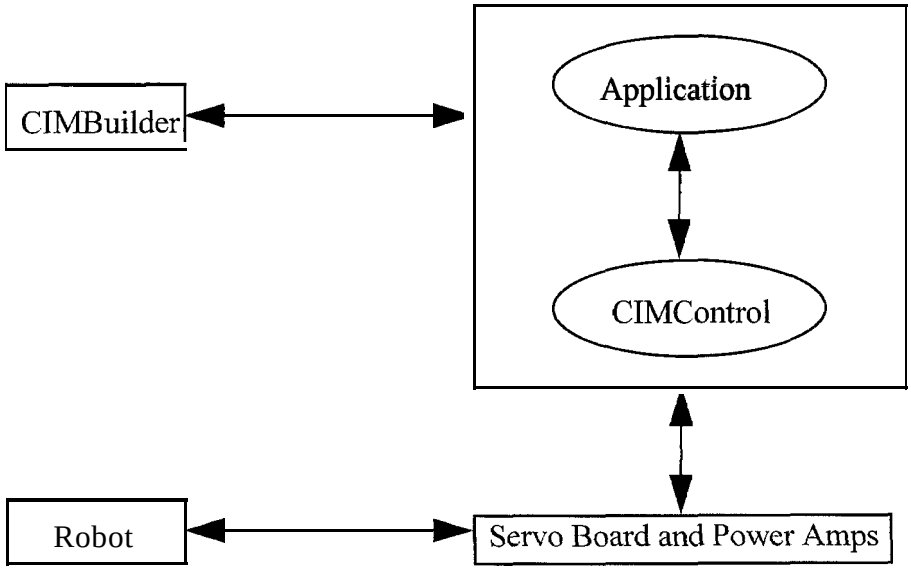

Figure 2. Real-time control of the robot

System Integration

Figure 3 shows how the pieces of the system work together. The remote site, typically using an inexpensive PC as its platform contains Cimetrix Open Architecture Controller software and White Pine teleconferencing software. The task is developed at the remote site and then sent to the local site for preview and execution. The local site contains the equipment (i.e., robot), Cimetrix Open Architecture Controller software running on any platform, and the White

Pine teleconferencing software. A camera views the work cell, sending the video to the remote site.

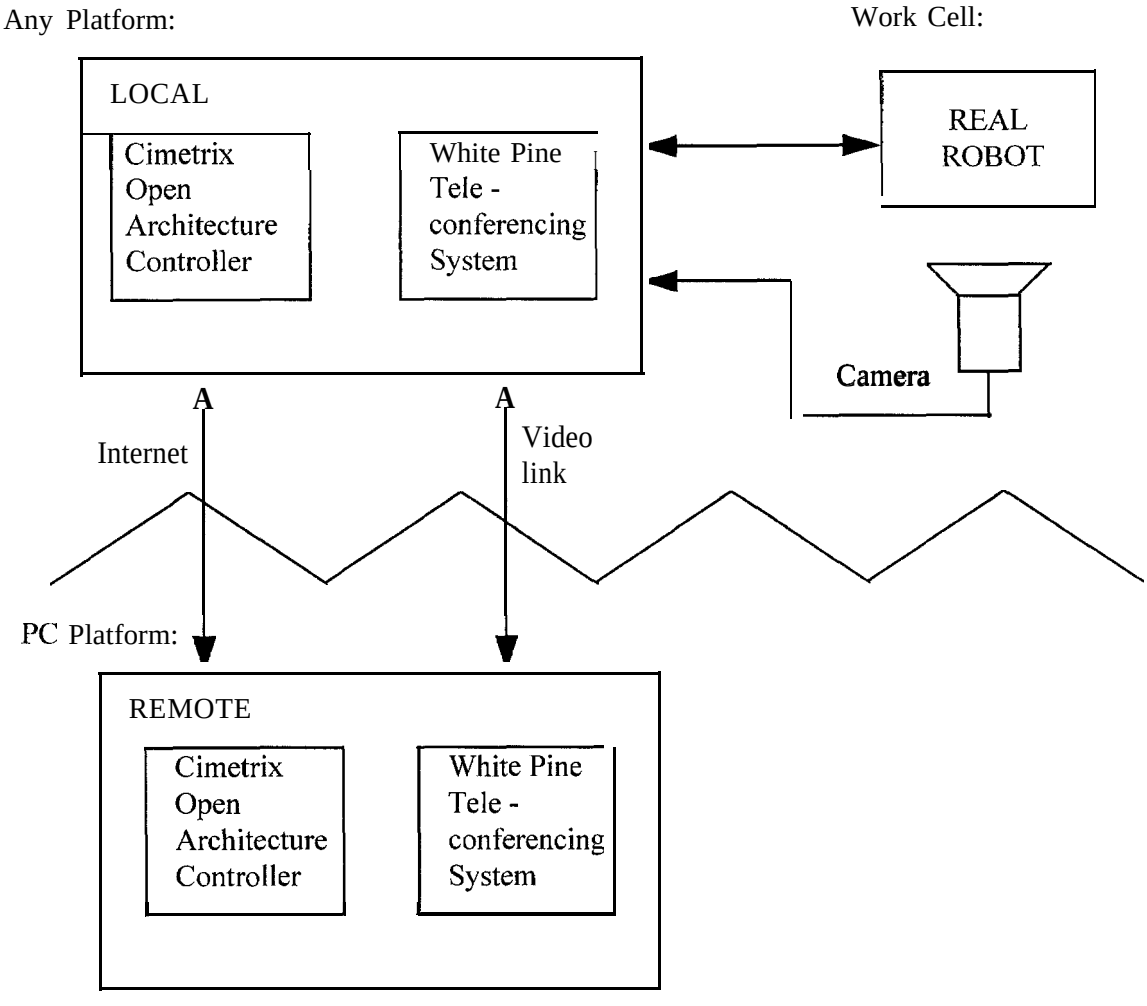

Figure 3. Low-Cost VCE System

Using the **System**

Once a work cell and equipment model have been developed using CIMBuilder, the actions are simulated, programmed and run. The data are then sent to the local site for program preview and final testing. When the local site is satisfied with the simulation the program is run. A video link provides feedback to the remote site.

Applications

Remote control of equipment has many applications. Such as:

- remote handling of contaminated waste.
- teaching factory workers how to use new equipment and simulate assembly plant actions.
- remote learning and use of equipment by rural communities, universities and public schools.
- remote control of equipment in harsh environments, e.g., space.

A space application of particular importance is the NASA Mars surveyor program.

NASA plans to send a series of spacecraft to Mars approximately every twenty-six months through the year 2005, beginning with the initial launch of two robotic explorers in November 1996. These robotic explorers are designed to study the composition of Mars, and to search for possible evidence of water. In future years, NASA envisions networks of more than a dozen of these explorers spread out over the entire planet. The Mars Surveyor Program will provide information NASA needs for the possibility of human flights to Mars, as well as create a blueprint for other planetary explorations. This program is expected to evolve as new technologies, like VCE, emerge and participate in these initiatives. VCE maybe accessed to test and preview motions of these robotic explorers to help assure capability prior to launch.

Suppose NASA scientists have received photographs of Mars, and would like to deploy a robotic explorer to search for carbonates, evaporates, or other minerals formed in the presence of water. By using CIMBuilder, the robot's task can be easily and rapidly developed using CODE. CIMBuilder is able to model the environment and terrain provided by the photographs, as well as the robots interaction with the environment, such as taking mineral samples or drilling into polar ice caps on Mars. This application would then be run in Cumulation, to graphically preview the motions of the robot. Meanwhile, NASA or a subcontractor could create a physical model of the environment. Once the remote and local users are satisfied with the simulation, the Cumulation program can be replaced with a real time control system, CIMControl, to control the actual mechanism. The video conferencing provides feedback to the remote user. Using VCE, other companies and universities collaborating in the Mars Surveyor Program may test their mechanisms, which NASA can incorporate into its mission.

The preliminary testing and motion monitoring may also optimize robot design. For instance, suppose that a design for a robot is not yet complete, or designers are not sure what the best design is for the robot task(s). VCE can test various virtual robots in different virtual scenarios, (or the same robot in different scenarios), allowing one to choose the best mechanism for its application. It can also help determine which robots currently in use would be best suited for certain applications, saving time and money.

'Future

From this base we are able to modify and add features. This includes the use of sensors, e.g., vision as part of the robot control system, an enhancement which is currently under development. This also includes the development of new scenarios and programs for possible use in the Mars Surveyor program and the development of new low level control algorithms, for example, a fuzzy logic based controller.

Conclusions

In this paper we described the feasibility of a low-cost VCE system. This was accomplished through the use of Cimetrix' integrated environment which allows a user to build a virtual environment and equipment, simulate the use of the equipment in the environment, and run the actual equipment. Cost goals were achieved by using a PC platform and inexpensive commercially available teleconferencing software.

References

[1] Unimate Puma Mark 11 Robot 500 Series - Equipment and Programming Manual, Unimation INC., Shelter Rock Lane, Danbury, Connecticut 06810

[2] Product literature, Cimetrix, Inc., 2222 South 950 East, Provo, Utah 84106,801-344-7000.

[3] Enhanced CU-Seeme Product Information, White Pine Software, INC., 1485 Saratoga Avenue, San Jose, California 95129-4934,408.446.1919

[4] Lumia, R., Todacheeney, M., Quintana, S., "Low-Cost Virtual Collaborative Environment Through Open System Technology," Proceedings of the World Automation Conference, Montpellier, France, May 1996

Acknowledgment

This work was supported in part by NASA under contract # NCCW-0087.