A Remote Robotics Laboratory on the Internet*

Y. U. Cao, T.-W. Chen, M. D. Harris, A. B. Kahng, M. A. Lewis and A. D. Stechert

Commotion Laboratory
UCLA CS Dept., Los Angeles, CA 90095-1596

Abstract

This paper describes ongoing work toward a remote robotics research site which allows repeatable remote experimentation on multiple mobile robots. Our system consists of ten small mobile robots hosting onboard Unix workstations. The robots provide facilities for sensing and moving obstacles, inter-robot positioning and communications, and user input. The Unix workstations allow the user to control the robots using common languages in a familiar environment, while also providing an interface to mass-market peripherals (secondary storage and vision), network access (telnet, FTP, mail, and HTTP), and robust multi-tasking. We believe that this work provides a foundation for future efforts toward new paradigms for remote research and user interaction with taskable hardware (e.g., colonies of application-specific robots). We envision applications in such domains as agriculture, environmental monitoring, and deep-space exploration.

1 Motivations

The UCLA Commotion (Cooperative Motion) Laboratory seeks new theories of cooperative task-solving, grounded in the field of robotics. One major goal of this research is to field colonies of robots at remote locations. These robots will be used to collect data, perform experiments, etc. They will be used by a wide variety of people including scientists, engineers, educators, and students.

We have identified several key problems that must be addressed. Among these are: communication between the experimenter and the remote colony, the remote tasking of robots, and the effective presentation of information to the user.

To address these problems, we have put together a robotic resource that will be made available to qualified users on the Internet within the next several months. Our resource consists of ten R3 research robots from IS Robotics, Inc., with added network connections, and is not currently duplicated anywhere in the world. We would enjoy sharing this infrastructure resource with other researchers and educators. In return, we expect to improve our understanding of how such resources can be shared and tasked.

We feel that the existing *information superhighway* infrastructure provides an ideal backbone for this project. While the information superhighway has demonstrated its use in the interactive sharing of information, our plan will add new capability by allowing control of and communication with physical, taskable resources. This technology will be enabling for such diverse purposes as: remote scientific investigation, continued widespread availability and affordability of scientific equipment, development of more rigorous standards for scientific discourse in highly experimental engineering disciplines, and Internet-based K-12 and undergraduate education. To help explain these visions for the future we describe three core ideas: taskable machines, colonies of taskable machines, and remote science.

1.1 Taskable Machines

A *taskable machine* is an *application-specific robot* designed to achieve only a narrow range of tasks within a restricted domain. The concept of a taskable machine is in direct contrast to that of a *universal robot* that possesses a wide range of capabilities. Since a taskable machine need not be functional in all possible environments and situations, the usual objectives of universal intelligence, massive computation, and powerful sensing can be deferred until necessary underlying technologies have matured. For robotics, the taskable machine concept is enabling: decades of effort toward universal robots have met with disappointment, while recent years have seen significant successes in application-specific robotic technology (particularly for defense [3], medicine [15, 32] and household applications [9]). [1]

Even though taskable machines are restricted in their application, their control is non-trivial. When a resource is constrained by its environment or its use (e.g., an agricultural pump or the Hubble space telescope), users seem to require little more than a scheduling interface. However, many scheduling problems are intractable, and the operation of remote hardware – particularly for sensing – entails challenging optimizations. [2] When mobility and unstructured environments are involved (e.g., a rover on Mars or on the ocean floor), there are even more challenging issues of operator interface, resource allocation, real-time feedback, and semi-autonomy. Software that acts as an autonomous safety net might be required to maintain system integrity when the network imposes communi-

*This research was supported in part by NSF Young Investigator award MIP-9257982 with matching support from High-Level Design Systems. The UCLA Commotion Laboratory is supported by NSF CDA-9303148 and matching funds from the UCLA School of Engineering and Applied Science.

[1] One analogy would be the contrast between application-specific integrated circuits (ASICs) such as controllers or DSP chips, and general-purpose microprocessors. While we view taskable machines as task-specific, other researchers such as Horswill [20] consider the notion of environment-specific robots.

[2] As examples: (1) Power and bandwidth limitations to telemetry may require intelligent image analysis before downlink from the sensing hardware. (2) Time delays and uncertainty in uplink of operator controls (e.g., to a deep-space probe via radio, or to a mobile robot via an unreliable link in the Internet) may require intelligent partitioning of control between the taskable machine and its remote operator. Cf. [1] for an overview of intelligent scheduling.

cation delays. [3]).

1.2 Colonies of Taskable Machines

Since each taskable machine has limited scope, achieving flexibility and sophisticated capability requires groups, or *colonies*, of taskable machines [2, 11]. In other words, future task-solving robotic systems will be made up of colonies of taskable machines that collectively achieve a wide range of tasks. Other motivations for robot colonies include:

- Building and using several simple taskable machines can be easier, cheaper, more flexible and more fault-tolerant than building a single universal robot to accomplish all possible tasks.[4]
- Tasks may be inherently too complex for a single robot to accomplish, or performance benefits can be gained from using multiple robots. A population of cooperating taskable machines is less limited by spatial and temporal constraints: certain problem classes are provably beyond the capability of a monolithic robot, essentially because a single robot cannot be spatially distributed (that is, in two places at the same time).
- Colonies of taskable robots are of independent research interest since their development may shed light on fields spanning the social sciences (organization theory, economics), life sciences (theoretical biology, animal ethology) and cognitive sciences (psychology, learning, artificial intelligence).

Historically, implementation of robotic colonies may have been viewed as strictly more difficult than implementation of a single robot. However, restricting robotic colonies to be composed of simpler application-specific robots can compensate for the added complexity of coordination. [16] gives example application domains. In recent years, works such as [31, 35] have established frameworks for task decomposition and control in multiple-robot systems. Several of these have been implemented using real robots (see [13] for a detailed survey).

1.3 Remote Science

In the future, educators and experimental scientists will be able to work with remote colonies of taskable machines via a "remote science" paradigm that allows:

- Spatial decoupling of experimenters and their experiments. This allows: (i) multiple users in different locations to share collaboratively a single physical resource (cf. notions of virtual "collaboratories" [41]), and (ii) enhanced productivity through reduced travel time, enabling one experimenter to participate in multiple, geographically distributed experiments.
- Sharing of expensive physical resources. In a climate of "big science" and constrained budgets, shared operation of scientific hardware is the only mechanism that can enable widespread availability and affordability of such equipment. Remote science helps level the playing field for researchers constrained by location or budget.
- Evolution of more rigorous standards for scientific discourse. Particularly in experimental engineering disciplines such as robotics, descriptions of experimental protocols can be vague, leading to irreproducibility of results. We believe that any mechanism for remote science transparently enforces experiment reproducibility since the remote researcher must completely specify his experiment prior to submitting it for remote execution.[5]
- Improved access of K-12 and undergraduate educators to leading-edge pedagogical material (cf. [26]). Again, an Internet-based remote science infrastructure will provide the means to escape from geographical and financial constraints. User interfaces built using HTML or similar graphical formats will facilitate access by casual or naive users.[6]

2 Toward an Implementation

The current state of our project, described in Section 3 below, makes ten mobile robots available for experimentation over the Internet. This is of interest given recent levels of activity in cooperative mobile robotics [13]. We provide the remote researcher with (i) access to a colony of near-leading edge research robots, (ii) software and hardware extensions to enable wireless networking and programmability in high-level languages, and (iii) infrastructure for remote experimentation. Thus, our project truly essays experimental remote science.

For present lessons to be more applicable to future efforts, and for relevance to "real science", we have centered on themes of mobility, autonomy, and remoteness in evaluating potential testbeds. Currently, we anticipate two future project phases which lead to unstructured outdoor environments and applications such as remote exploration, geological and botanical sampling, and ecosystem monitoring.[7] The three project phases, and their relationships to each other, may be summarized below.

Phase One (in progress) provides an indoor experimental facility allowing users from cooperating institutions to log in and control our robotic colony via a World-Wide Web (WWW, or Web) interface. Here, we believe users will focus on research issues in cooperative mobile robotics. Initial functionality includes a full Unix development environment (X, gcc, Emacs, gdb, etc.), Web interface, simple scripting capability, robot control via a shared-memory architecture, and full Internet connectivity (TCP/IP, SMTP, FTP, etc.).

[3] See [8], which describes a safety net for mobile robots based on potential fields, and uses the term "teleautonomy". Other useful references are contained in the substantial literatures on teleoperation and telerobotics.

[4] Note that: (1) A single robot translates to a single point of system failure. (2) Taskable machines by their nature do not require the riskier leading-edge technology needed by universal robots. (3) A system of multiple robots can provide graceful degradation under failures.

[5] Sensing, actuation, program and video traces which comprise feedback to the remote experimenter will at the same time provide back-annotation of the experimental protocol.

[6] Public policy considerations must also be addressed, e..g., how should access to a space telescope be prioritized between a local researcher and a remote researcher, between a school assembly and a Ph.D. student's experiment, etc.?

[7] We are not as interested in a "remote undergraduate biology or chemistry laboratory" (cf. [28]). Requirements such as manipulation, machine vision, safe operation in human environments, etc., dominate such a concept; these are not so dominant in the distributed sensing, monitoring, and exploration tasks that we envision for colonies of mobile taskable machines.

Phase Two will introduce behaviors (i.e., task primitives, high-level tasks) into the colony via a layered control approach. Of highest priority are behaviors which preserve the safety and integrity of the colony (see [29] as well as the discussion of requirements and implementation issues in the remainder of this section). This "safety net" will provide protection against errant or harmful user programs, against hardware failures and power loss, etc. At the same time, the robot arena will be enhanced with hardware to allow manipulation of large obstacles, thus affording more complex experimental environments and the removal of failed robots. Given this increased flexibility and programmability, the sophistication of experiment design will increase; we will provide more extensive data logging (sensing, actuation, variable-resolution or filtered on-board video), and experiment profiles. Finally, the interface will improve, following the development of HTML3 and other Web facilities.

Phase Three will field equipment in an outdoor setting, using heterogeneous (legged or possibly aerial) robot platforms which are almost fully autonomous. Experiments will last on the order of days to weeks, and will be run remotely over the Internet. Science (e.g., botany, soil science, and atmospheric chemistry) will be performed, with task complexity hopefully on the order of that achievable by the JPL Pathfinder Mars rover [23]. We believe that an important benefit will be near-continuous, close-in monitoring for regions in which human access is too costly.

Throughout, our system will support rigorous scientific experimentation by enforcing well-defined experimental protocols via its interface. Note also that our Phase Three does not entail teleoperation or telepresence; such approaches are inappropriate for reasons including network failure and the high cost of continuous human operation. Rather, our goal is to extend the robots' sensing and manipulation capabilities to optimize information gathering and delivery to the user. The remainder of this section lists functional requirements for our system and contrasts them with capabilities afforded by existing efforts.

2.1 System Requirements

A taskable robotic system for remote experimentation has the following requirements for mobility, sensing, and data logging, real-time response, communication, user interface, and programming support.

- **Mobility** implies the following technical problems. (i) **Energy.** Mobile robots must carry their own energy sources or must derive their energy from the environment using, e.g., solar cells. If the robots are dependent on an external source of energy, then they must be able to navigate safely to a refueling station when necessary. (ii) **Location.** To exploit maps of the environment, or to navigate to prescribed sites, robots must determine their positions; possibilities include differential Global Positioning System (absolute coordinates), beacon system (relative coordinates), or landmark-based navigation (in unstructured environments). Robots must also be able to determine locations of other objects relative to their own frame of reference. (iii) **Collision avoidance.** Robots must detect both static and moving objects in the environment and execute avoidance algorithms.

- **Data logging** must be sufficient for the scientific application at hand (e.g., remote monitoring or exploration), as well as for reconstruction and analysis of the scientific experiment itself. For the former, on-board data storage is required along with possibly intelligent data/image processing prior to downlink in power- or bandwidth-limited scenarios. For the latter, low-level sensor and actuator traces, and perhaps (compressed, low-sample rate) video and program traces should be archived for eventual transmission to the remote experimenter.

- **Real-time response** is required since the robots are mobile in possibly hazardous environments. On-board requirements include a fast CPU and an approximation of real-time preemptive multitasking. Remote user requirements include low-resolution feedback of sensing and actuation (possibly including some form of video feedback). Since such feedback has multiple sources with varying bandwidth requirements, flexibility in the spatial or temporal resolution of the feedback streams must also be possible.

- **Communication** must be sufficient for the agents in the robot colony to coordinate their actions. There is a tradeoff among communication bandwidth, sensing and processing, in that communication can be achieved implicitly by monitoring other robots, by sensing the effects of other robots' actions, or most directly through explicit messages that are transmitted over some communication channel [13]. If robots are able to model other robots (i.e., beliefs, states, goals, etc.), then communication requirements will be correspondingly decreased. For our purposes, standard wireless networking technologies provide the necessary bandwidth and flexibility in topology. Communication must also enable telemetry (uplink of operator control and downlink of scientific data). As noted above, real-time downlink might include behavioral state and low-level sensory data. If video feedback is desired, it will dominate the downlink requirement and bandwidth on the order of hundreds of Kbps will be necessary.

- The **user interface** must provide facilities for user validation and acquisition of the taskable resource, program submission, experiment registration, experiment setup (i.e., to achieve repeatably a prescribed initial configuration of robots and obstacles), on-line control of the experiment, viewing of near real-time feedback (e.g., the low-resolution video and sensor traces noted above), and receiving/viewing of the experimental data (e.g., in the form of a report). Additionally, tutorial facilities may be offered. For universal accessibility, the interface should be graphical and Web-based.

- **Programming support** should be on at least three levels: Level 1 allows interpreted, immediate execution; Level 2 allows scripts of Level 1 instructions; and Level 3 affords high-level language support and access to hardware via an application programmer's interface (API). Simulation facilities can speed experiment design and programming, as well as off-load demands on the actual robotic resource when the user community is large. Particularly, with sophisticated experiments that require Level 3 support, the API

Figure 1: The UCLA/ISR R3+ robot.

should allow both scripting capability and low-level access to the robots.[8] An ideal solution would be Unix-based with a standard mechatronic interface [34] to low-level control of the robot. Such an approach would leverage the mature Unix development environment, affording scripting and rapid prototyping capabilities, while retaining low-level hooks into the hardware.

2.2 Contrasts With Previous Work

Clearly, our goal of implementing Web-accessible autonomous mobile robots entails solving many problems. A brief digression into an operating systems analogy points out transferable techniques and enables a more tractable picture of the system design. First, the *shell* provides a user interface through which users submit commands and receive responses from the system. Basically, the shell provides user registration, user login, utilities such as job submission commands/languages, data logging commands/languages, commands "who" and "talk" to facilitate collaboration. The shell also provides a programming environment (compiler, simulator, and debugger). Second, the *kernel* schedules resources[9] and implements the shell commands. Thus, the kernel is responsible for such system aspects as: (i) communication between entities, (ii) sharing, scheduling and partitioning of resources, (iii) the safety net, (iv) maintaining information about users (e.g., to enforce different levels of access), and (v) security[10].

With this in mind, even a minimal realization of our system must accomplish the following four steps: (i) realize remote operation of taskable machines, (ii) add programmability, (iii) add multiplicity and autonomy, and (iv) add collaborative remote experiment capability.

Regarding remote operation of taskable robots (our step one), the field of telerobotics has a 50-year history [38] and is a form of resource sharing. When we add the goal of Web-accessibility, relevant previous works include the following.[11]

- The Mercury project at USC [19, 21] consists of a robot manipulator moving above a section of earth known to contain artifacts. There is a downward-pointing camera attached to the distal end of the robot arm. By clicking on an image or buttons on the Web page, users can make the robot move along the X-Y axes and the camera along the Z axis. In real-time operation, GIF images of the camera view are updated on the Web page, and a session is logged as an MPEG movie;

[8] For example, [21] showed that by using a Web interface, it is possible to make robotics applications widely accessible by appropriately abstracting the interface. On the other hand, [18] showed that low-level real-time control can also be achieved over the Internet.

[9] For users, resources include actuation, sensing, computation and communication capabilities of the taskable hardware, which form the basis for the remote experiment. For the taskable hardware, resources include space, communications media, etc. and engender many resource-conflict issues.

[10] By "security", we mean "protection", i.e., "a mechanism for controlling the access of programs, processes or users to the resource defined by a [computer] system" [39].

[11] Other Web applications include: (1) Net-frog [28] uses QuickTime movies to demonstrate dissections in the frog anatomy. Users use a mouse to practice dissection on still images of frog; the program monitors and notifies the users if, e.g., they try to cut at the wrong place. (2) The Monterey BayNet project [12] establishes the backbone for multiple educational initiatives and research projects. ATM, Frame Relay, and ISDN networks are interconnected, with the Multicast Backbone [40] and World-Wide Web providing tools for complete connectivity to a wide variety of existing information sources. Ongoing projects include the live exploration of Monterey Canyon using a deep-sea remotely-operated vehicle, and a "virtual telescope" interface to astronomical data at The Monterey Institute for Research in Astronomy (MIRA). (3) The NERO project [26] is an ongoing multiple-institution effort in Oregon to create a virtual classroom, i.e., collaborative research and teaching among the member institutions and with industrial sites. ATM provides the basic network technology. Collaborative tools, distance learning and real-time control applications such as robotics and flight simulators are some of the research areas.

the only sensory data is the energy level. This project has been decommissioned since March 31, 1995.

- The Australia's Telerobot On The Web project at the University of Western Australia [24] challenges users to control the parallel-jaw grippers of a six-axis manipulator to pick up blocks on a table. Through a fill-out form, users can submit desired (X,Y,Z) coordinates, and roll,pitch,yaw actions. A fixed camera gives a side view of the arena, and the image is updated when the user command is submitted and executed.

- The Bradford Robotic Telescope [14, 25] allows users to work either in batch mode[12] or in real-time mode. The user describes what he wishes to observe and the telescope moves accordingly. One of the strong features of this project is the good implementation of user access control: users must register, login (passwords can be changed via e-mail), and submit jobs. Furthermore, jobs are prioritized; only those with highest priority are allowed real-time control of the telescope.

All three of these projects afford remote manipulation of taskable hardware through a Web interface. In this sense, they have demonstrated the feasibility of certain basic aspects of our project plan. However, in each of these projects there is only one piece of taskable hardware (the Mercury project uses an IBM manipulator, the UWA project uses an ASEA industrial robot, and the Telescope project uses a telescope interfaced to PCs); hence, issues of multiplicity and autonomy do not arise. Furthermore, since there is no programmability involved, neither programming environment nor data logging is an issue in these projects.

Regarding adding programmability (our step two), the most significant work we know of is the Onika project at Carnegie-Mellon University [18], which supposedly provides a "virtual laboratory / virtual factory".

- Onika is an interface to a real-time operating system called Chimera which supports reusable real-time software. Through Chimera, reusable software modules downloaded from other sites can be immediately assembled. Onika's contributions are (i) an iconic interface for linking software modules (basically, like an object manager in an object-oriented environment), (ii) the ability to follow hyperlinks to retrieve remote software modules[13], and (iii) display of video and other information at run time. In a demonstration, remote and local software modules for controlling a Puma manipulator were linked and executed using the Onika interface, with real-time video of the Puma shown on the interface.

For our purposes, something similar to the Chimera operating system would be able to provide a good software development environment for collaborating sites, and a friendly user interface like Onika would be equally useful.[14]

We know of no works that address the last two steps – multiplicity, autonomy, and collaboration – of our minimal system realization. Certainly, computer-supported collaborative design [41] and "Virtual Environment" efforts[15] are well-developed research fields, and it is possible that techniques from those fields will allow collaborative remote experiment capability to be built on top of realizations of the first three steps above.

3 Current System

The system is now in the early stages of implementation and it is called the W3R3 Project. Ten R3 robots have previously been acquired and an 8 meter x 7 meter arena has been set aside for remote experimentation. The following components are currently in place: (i) IS Robotics R3 robots supplemented by Unix workstations with Internet access, (ii) a Global Positioning System, (iii) battery recharging support, and (iv) a Web-based user interface. These features are described in the following subsections.

3.1 Robot Platform

Based on previous experience, we believe that the R3 is well suited to the project requirements outlined in section 2. However, two caveats apply: (i) they have no interface to the Internet, and (ii) they have insufficient GPS resolution (i.e., the measured spatial resolution of the R3's original GPS subsystem was 10 cm with a time resolution of 5 seconds).

As initially configured, the R3 is a small, autonomous mobile robot about 30 cm both in diameter and in height (see Figure 1). Computational requirements are satisfied by 68332 and 68HC11 microcontrollers. The R3 runs the Venus operating system [27] and runs user programs in 1MB of non-volatile RAM.

The sensor array of the robot includes infrared proximity sensors, bump sensors, shaft encoders, power status indicators, and a user input panel. The robot moves using a differential drive and can manipulate objects using a force-sensing gripper subsystem. Power is supplied by rechargeable NiCad batteries. The robot has the capability to detect low power conditions and recharge under software control when attached to an external power source.

3.2 Computation Upgrade

A key aspect of our design is the on-board support of standard Internet protocols (e.g., TCP/IP, FTP, etc.). We examined several ways of modifying the R3 to accept network connections including direct modification of Venus. An alternative was to add another

[12] I.e., submitting a job and receiving an image result – sometimes days later.

[13] Because network delays are relatively long and unpredictable, the remote software modules are first downloaded automatically by Onika, then the whole program executes on the machine where all modules reside.

[14] The Onika interface is built on top of X-windows; it is not as simple to achieve an iconic interface and real-time video using existing Web facilities.

[15] Various research works toward "Virtual Environments" and "Virtual Reality" have already produced a number of software packages, architectures, and operating systems. Two examples are as follows: (i) The VEOS (Virtual Environment Operating Shell) project [10], which is a management facility for generation of, interaction with and maintenance of virtual environments. An early demonstration in a blocks world allowed four participants to independently navigate and manipulate movable objects in a shared virtual space. (ii) The work of [30] allows informal collaboration by embedding an information retrieval tool (Gopher) to a "text-based virtual reality" environment; this is an example of merging computer-mediated conferencing and online information retrieval.

processor capable of running Unix, which supports network communications natively and to write software that interfaces the two. In addition to satisfying our original requirements, choosing the latter option also allowed us to significantly upgrade the computational capability of the R3 and provide access to the large base of Unix applications.

Therefore, our robots have been augmented with a compact 486DX2-66 Unix workstation, based on the PC/104 bus standard, with 4MB of RAM and .5GB mass storage. The workstations run a variant of Unix called Linux, for which source code is available, making kernel-level modifications possible. This ability becomes important when device drivers have to be written for task-specific sensors or actuators. The entire workstation fits in a 12 cm cube that is secured to the top of the R3.

The Unix workstation and the robot communicate via virtual shared memory. By this, we mean that no actual physical memory is shared; instead, we emulate shared memory with software that drives RS-232 hardware linking the two computational resources. The shared memory is logically implemented as files on the Unix workstation that store the current values of the robot's sensors and the current set points of the robot's actuators. The update frequency between the workstation and the R3 is currently 32 Hz.

The advantage of this file-based shared memory approach is that closed loop control of the robot can be implemented as a high priority Unix process that reads the sensors file, computes new actuator set points, and then updates the actuators file. Since virtually all languages support some kind of file access, this system provides nearly universal access to programming languages.

3.3 Gobal Positioning System

The GPS consists of four overhead cameras with slightly overlapping fields of view. The camera images are digitized and processed to yield the position and orientation of each robot and of obstacles in the environment, with updates occurring at 30 Hz with a spatial resolution of about 3 cm. The system allows queries of the robot's own position, of other robots' positions, and of obstacles' positions. To provide this information to processes running on the robots, sockets are used to transfer data from the off-board vision processing workstation to the robots local filesystems.

3.4 Recharging

Battery recharging support is accomplished by "feeding trough" technology. When a robot needs to recharge, it proceeds to a wall on one side of the arena where two parallel horizontal plates maintain enough potential to drive the robot's recharging circuitry. Physically, the robot's grippers have contacts above and below that are positioned so that the fingers can be inserted between the parallel plates to allow recharging. This allows the current system to run indefinitely. With power-conscious safety net software, robots can interrupt the current experiment under low power conditions and recharge before completing the experiment.

3.5 Communication

The communication between basestation and each R3 is discussed as follows in terms of OSI layers.

At the physical layer, a pair of Proxim radio modems is dedicated as a wireless serial link between each R3 and its basestation. The link can be viewed logically as a standard RS-232 cable, with a data rate of 19.2 Kbps. Based on spread-spectrum technology, the radio bandwidth is multiplexed over seven radio channels. Each modem can select one of the channels to transmit or receive data, but only four radio channels can be concurrently used in the same area. To support more radio connections, each radio channel can be further divided into 64K logical sub-channels. Contention for the sub-channels is handled at the MAC (medium access control) layer by using the CSMA/CA (carrier-sense multiple access with collision avoidance) protocol. However, due to the inherent delay caused by CSMA/CA, throughput degrades as the number of active sub-channels increases.

In our configuration, we have used the maximum of four independent channels, statically assigning radio modems to sub-channels to accommodate more than four simultaneous connections. We are currently running PPP (Point-to-Point Protocol) on the wireless links to provide TCP/IP functionality. The basestation therefore can communicate with the Unix workstation on-board each R3 by using any of the facilities of the TCP/IP protocol suite. At the same time, the basestation has an Ethernet connection to the Internet, which provides high bandwidth connectivity to other Internet hosts.

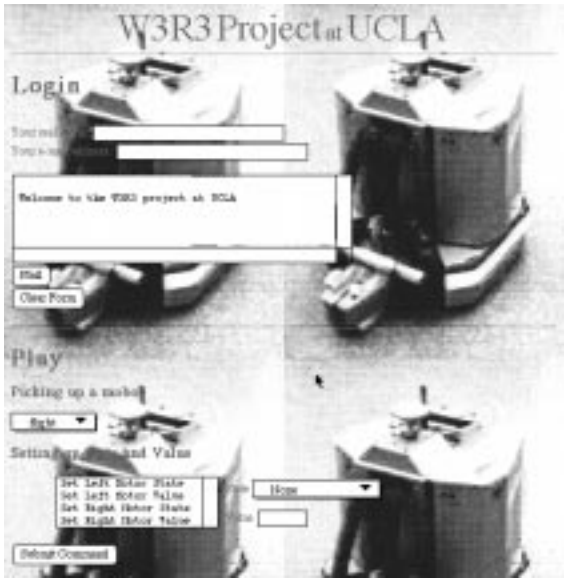

Figure 2: Part of the Web page.

3.6 User Interface

On the basestation, an NCSA HTTP server runs and provides a graphical user interface (GUI) to the mobile robots.

In order for the users to interact effectively with the mobile robots in the lab, it is necessary that the GUI must support the following capabilities: (i) submission of basic commands, e.g., Forward, RaiseGripperLift, (ii) viewing of actuator and sensor states, and (iii) submission of program files. In addition, the following are highly desirable: (iv) video of the lab overview, and (v) real-time animation of sensory readings.

We implement our GUI based on Web facilities (i.e. HTTP, HTTP server, Common Gateway Interface, HTML, and Web browser). Web facilities are chosen as the implementation language since (i) Web facilities are widely available and therefore maximize

the accessibility of our system, and (ii) Web facilities are rapidly evolving so that in the near future we will be able to implement our entire GUI based only on Web facilities. Figure 2 shows part of the current Web page.

Since the current Web facilities cannot support all the features needed for our interface,[16] we use a hybrid approach for now, and plan to replace non-Web-based parts as Web facilities improve.

The GUI can be divided into 3 parts: (i) registration/login, (ii) programming, and (iii) display.

The registration/login part is performed in a manner similar to the Bradford telescope project. We are adding new capabilities for users to schedule experiment time slots with our lab.

Programming is supported at different levels:

- At Level 1, users can set the state and value of actuators on a particular R3.
- At Level 2, users can program in a script language to make an R3 to do simple tasks like moving in the figure of a square. A script is composed of lines with the format <command time-interval>. Commands currently implemented include Forward, Backward, FastForward, FastReverse, TurnLeft, TurnRight, Pause, RaiseGripperLift, and LowerGripperLift.
- At Level 3, users can submit a program to the basestation, and get possible error messages back on a Web page. Since there is no Web facility for uploading files to an HTTP server, users must e-mail the program. Additionally, very trusted users can simply telnet into the basestation.

The display part of the GUI transmits video and sensory data in semi real-time. In the current setup, we send video stream via a video conferencing tool called nv (NetVideo) [17]. This requires that the client has nv capability, and is therefore restrictive. Sensory data are plotted and displayed in primitive animation using Netscape's server push method. Finally, the user can request that the complete log of sensory readings be sent to him/her via e-mail.

4 Summary

We have described ongoing work toward a remote robotics research site which allows repeatable remote experimentation on multiple mobile robots. Our system consists of multiple mobile robots hosting onboard Unix workstations, and approximates the Phase One functionality described in Section 2 above. As we move toward the ability to conduct "real science" via networked, remote colonies of taskable robots, we envision applications in such domains as agriculture, environmental monitoring, and deep-space exploration.

References

[1] M. Aarup, M. Zweben and M. S. Fox, eds., *Intelligent Scheduling*, Morgan Kaufmann, 1994.

[2] A. Agah, *Sociorobotics: The Study of Robot Colonies From Simulation to Hardware Realization*, Ph.D. thesis, USC CS Dept., 1994.

[3] ARPA, working notes of *ARPA Taskable Machines Workshop*, Feb. 15-16, 1995.

[4] T. J. Berners-Lee, R.T. Fielding, and H. F. Nielsen, "HyperText Transfer Protocol – http1.0", *Internet draft*, Mar. 1995, http://www.ics.uci.edu/pub/ietf/http/draft-ietf-http-v10-spec-00.txt.

[5] T. J. Berners-Lee, R. Cailiau, J. F. Groff, and B. Pollermann, "World-Wide Web: The Information Universe", *Electronic Networking, Research, Applications and Policy*, No.2, 1992, pp.52-58.

[6] T. J. Berners-Lee and R. Cailliau, "World-Wide Web: Proposal for a Hypertext Project", 1990, http://info.cern.ch/hypertext/WWW/Proposal.html.

[7] T.J. Berners-Lee and D.Connolly, "HyperText Markup Language Specification - 2.0", *Internet draft*, Feb. 1995, http://www.osn.de/htmlspec/HTMLSPEC.html.

[8] J. Borenstein and Y. Koren, "Teleautonomous Guidance for Mobile Robots", *IEEE Trans. on Systems, Man and Cybernetics*, 20(6), Nov.-Dec. 1990, pp. 1437-1443.

[9] A. Branch, *personal communication*, Oct. 1994.

[10] W. Bricken and G. Coco, "The VEOS Project", *Technical Report*, Tech Report R-93-3, Univ. of Washington, 1993.

[11] R. A. Brooks and A. M. Flynn, "Fast, Cheap and Out of Control: A Robot Invasion of the Solar System", *Journal of the British Interplanetary Society*, 42(10), Oct. 1989, pp. 478-485.

[12] D. Brutzman, "Networked Ocean Science Research and Education, Monterey Bay California", *Proc. INET'95*, 1995.

[13] Y. U. Cao, A. S. Fukunaga, A. B. Kahng and F. Meng, "Cooperative Mobile Robotics: Antecedents and Directions", *Proc. IEEE/RSJ Intl. Symp. on Intelligent Robotics and Systems*, Aug. 1995, to appear.

[14] M. J. Cox and J. E. F. Baruch, "Robotic Telescopes: An Interactive Exhibit on the World-Wide Web", *Proc. 2nd International World-Wide Web Conference*, Oct. 1994.

[15] K. G. Engelhardt, *personal communication* (talk at Los Angeles Area Robotics and Automation Symposium), May 1995.

[16] D. E. Franklin, A. B. Kahng and M. A. Lewis, "Distributed Sensing and Probing With Multiple Search Agents: Toward System-Level Landmine Detection Solutions", *Proc. SPIE Aerosense-95: Detection Technologies for Mines and Minelike Targets*, Apr. 1995.

[17] R. Frederick, "Experiences with Real-Time Software Video Compress", Jul. 1994, ftp://parcftp.xerox.com/pub/net-research/nvpaper.ps.

[18] M. W. Gertz, D. B. Stewart, and P. K. Khosla, "A Human-Machine Interface for Distributed Virtual Laboratories", *IEEE Robotics and Automation Magazine*, Dec. 1994, pp. 5-13.

[19] K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, and J. Wiegley, "Desktop Teleoperation via the World-Wide Web", *Proc. IEEE Intl. Conf. on Robotics and Automation*, 1995.

[20] I. Horswill, *Specialization of Perceptual Processes*, Ph.D. thesis, MIT EECS Dept., May 1993.

[21] http://cwis.usc.edu:80/dept/raiders.

[22] http://hoohoo.ncsa.uiuc.edu/cgi/overview.html.

[23] http://mpfwww.jpl.nasa.gov/ jwright/fact_sheet /fact_sheet.html.

[24] http://telerobot.mech.uwa.edu.au.

[16] Currently widely used Web facilities are HTTP 1.0 [4], HTML 2.0 [7], CGI 1.1 [22], and the most widely used browser is Netscape (Version 1.1 as of mid-April 1995). However, the current Web facilities do not effectively support: (i) real-time animation (although "poorman's animation" is achievable via, e.g., Netscape 1.1's "server push" and "client pull"), and (ii) file upload capability. We note that adding file upload function into HTML fill-out forms has been proposed [33, 36], and has been implemented experimentally by some browsers such as Hot Java.

[25] http://www.eia.brad.ac.uk/rti.

[26] http://www.nero.net.

[27] IS Robotics, *Venus Reference Manual*, 1995.

[28] M. B. Kinzie, V. A. Larsen, J. B. Burch, and S. M. Boker, "Net-frog: Using the WWW to Learn about Frog Dissection and Anatomy", *Proc. INET'95*, 1995.

[29] M. A. Lewis, A. H. Fagg, and G. A. Bekey, "The USC Autonomous Flying Vehicle: An Experiment in Real-Time Behavior Based Control", *Proc. of IEEE Intl. Conf. on Robotics and Automation*, Vol. II, May 1993, pp. 422-439.

[30] L. Masinter and E. Ostrom, "Collaborative Information Retrieval: Gopher from MOO", *Proc. INET'93*, 1993.

[31] M. Mataric, *Interaction and Intelligent Behavior*, Ph.D. thesis, available as Technical Report AI-TR-1495, MIT AI Lab, Aug. 1994.

[32] B. L. Musits, W. L. Bargar and E. J. Carbone, "Image-Driven Robot Assists Surgeons With Total Hip Replacements", *Industrial Robot* 20(5), 1993, pp. 12-14.

[33] E. Nebel and L. Masinter, "Form-Based File Upload in HTML", *Internet draft*, Apr. 1995, ftp://ietf.cnri.reston.va.us/internet-drafts/draft-ietf-html-fileupload-02.txt.

[34] E. J. Nicolson, "Standardizing I/O for Mechatronic Systems (SIOMS) Using Real Time UNIX Device Drivers", *Proc. IEEE Intl. Conf. on Robotics and Automation*, May 1994, pp. 3489-3494.

[35] L. E. Parker, *Heterogeneous Multi-Robot Cooperation*, Ph.D. thesis, MIT EECS Dept., Feb. 1994.

[36] D. Raggett, "HyperText Markup Language Specification Version 3.0", *Internet draft*, Apr. 1995, http://www.hpl.hp.co.uk/people/dsr/html3/CoverPage.html.

[37] K. S. Roberts, "Robot Active Touch Exploration: Constraints and Strategies", *Proc. IEEE Intl. Conf. on Robotics and Automation*, 1990, pp. 980-985.

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

[39] A. Silberschatz and P.B. Galvin, *Operating System Concepts*, Addison-Wesley, 1994.

[40] A. Thyagarajan, S. Casner, and S. Deering, "Making the MBone Real", *Proc. INET'95*, 1995.

[41] I. Tou, S. Berson, G. Estrin, Y. Eterovic, and E. Wu, "Strong Sharing and Prototyping Group Applications", *IEEE Computer*, 27(5), May 1994, pp.48-56.