Desktop Teleoperation via the World Wide Web*

Ken Goldberg, Michael Mascha, Steve Gentner, Nick Rothenberg,
Carl Sutter, and Jeff Wiegley
University of Southern californiat

Initiated at CERN in 1992, the "World Wide Web" (WWW, including the HTML language and the HTTP protocol) [1] provides a standard graphical interface to the Internet. "Point-and-click" clients for reading hypertext have been ported to most computer platforms; the worldwide number of users is well over 500,000 and growing rapidly.

As feasibility study, we built a system that allows a robot manipulator to be teleoperated via the WWW. Although the $eld of teleoperation dates back over 50 years, the WWW provides a low-cost and widely-available interface that can make teleoperated resources accessible to anyone with a desktop (or laptop!) computer and modem.

The "Mercury Project" consists of an industrial robot armJitted with a CCD camera and a pneumatic system. We placed a sandbox$lled with buried artifacts in the robot workspace. Using the ISMAP feature of HTTP, users can remotely move the camera to view desired locations or direct a short burst of compressed air into the sand to view the newly cleared region.

To our knowledge, the Mercury Project is the Jirst system to permit WWW users to remotely view and alter *the real world. Since it came online September 1, 1994, the system has been available almost continuously. As of February 1, 1995, the project had been accessed by over 50,000 unique sites around the world*[1].

This paper focuses on interface design, robot hardware, and system architecture. Archival information including example images, operator logs and the answer to the puzzle is available at:

http://www.usc.edu/dept/raiders/

1 Goals sf the Project

In the Spring of 1994, hundreds of WWW servers were coming online every week. We conjectured that it might be possible to use this medium to allow low cost public access to a teleoperated robot, in effect providing: *desktop teleoperation.*

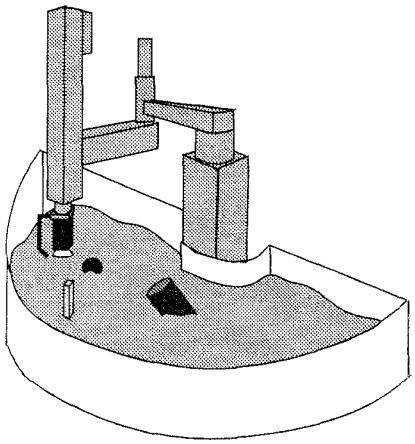

Figure 1: Robot, camera and air nozzle above workspace.

As illustrated in Figure 1, we set up a SCARA-type robot arm over a semi-annular workspace containing sand and buried artifacts. We attached a CCD camera to the end of the arm along with a nozzle to direct air bursts into the sand. We then developed an interface so this hardware could be controlled via the WWW.

Our primary criterion was that the system be reliable enough to operate 24 hours a day and survive user attempts at sabotage. A practical criterion was that the system be low in cost as we had a limited budget. It is worth noting that the manufacturing industry uses similar criteria, reliability and cost, to evaluate robots for production. Thus our experience with RISC robotics [2] proved helpful.

Our secondary **goal** was to create **an** evolving WWW site that would encourage repeat visits by users. Towu-d this end, all of the buried artifacts were derived from an ucnamed 19th Century text. Users are challenged to identify this text and thereby collectively solve the "puzzle". After tach 5-minute operating session, users are prompted to describe their findings and hypotheses in an ongoing Operator's Log. As of 1 February 1995, although the Log includes over **1000** pages of entries, the puzzle has yet to be solved.

'This work was supported in part by NSF Young Investigator Award IRI-9457523 to Prof. Goldberg, who can be reached at goldberg@usc.edu or **(+213)** 740-9080. A very early version of this paper was presented at the Second International WWW Conference, Chicago, IL. Oct 17, 1994.

'~oldberGentner, and Wiegley are with the Computer Science Department, Mascha and Rothenberg are with the Anthropology Department, Sutter is with the Center for Scholarly Technology;

The Mercury Project will be decommissioned in March 1995 to prepare for a new project.

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

2 Related Work

Goertz demonstrated one of the first "master-slave" teleoperators 50 years at the Argonne National Laboratory[3]. Remotely operated mechanisms have long been desired for use in inhospitable environments such as radiation sites, undersea [4] and space exploration [5]. At General Electric, Mosher [6] developed a complex two-arm telwperator with vidw cameras. Prosthetic hands were also applied to teleoperation [7]. More recently, teleoperation is being considered for medical diagnosis [8], manufacturing [9] and micromanipulation [lo]. See Sheridan [11] for an excellent review of the extensive literature on teleoperation and telerobotics.

Most of these systems require fairly complex hardware at the human interface: exoskeleton master linkages are attached to the human arm to minimize the kinesthetic effects of distance to create a sense of "tele-presence". Our objective was to provide widespread access by using only the "point-and-click" interface available under the standard HTML language.

A number of WWW sites provide access to remote devices such as cameras, coffee pots, and coke machines 1121. Although we believe our system was the first to allow WWW users to manipulate a remote environment, remote motion control was independently explored by several other researchers. In October 1994, Mark Cox of Bradford University reported a system that allows WWW users to remotely schedule photos from a robotic telescope [13] and Rich Wallace of NYU demonstrated a remote camera that can be selectively aimed using a **WWW** ISMAP [14]. Shortly after the Mercury Project came online, Ken Taylor of the University of Western Australia demonstrated a remotely controlled six-axis telerobot with a fixed observing camera 1151. Although Taylor's system requires users to type in spatial coordinates to specify relative arm movements, his system allows WWW users to pick up blocks by controlling the robot's parallel-jaw gripper.

3 System Design and User Interface

To facilitate use by a wide audience of non-specialists, we sought to make all robot controls available via the standard point-and-click mouse commands as shown in Figure 2. This forced us to consider a 2D workspace with only a few buttons for out-of-plane effects. Users are trained with an on-line tutorial prior to operating the robot.

The user interface centers around the bitmap that we call the "status image" as shown in Figure 3. Any number of "observers" carl simultanwusly view the status image, but only the current "operator" can send commands by clicking on the image. To limit access to one operator at a time, we implemented password authentication and a queue that gives each operator 5 minutes at the helm.

When the operator clicks on the status image using the mouse, the XY coordinates are transferred back to our server, which interprets them to decode the desired robot action. This action can be: (1) a global move to center the camera

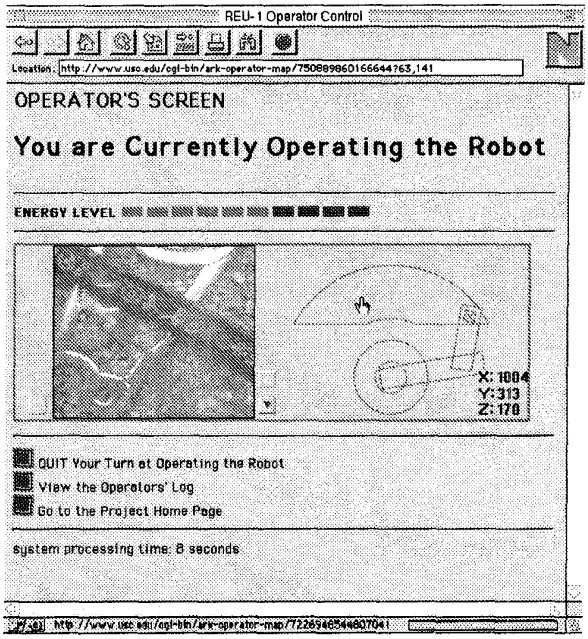

Figure 2: The interface as viewed by a WWW browser.

at XY in the schematic workspace, (2) a local move to center the camera at XY in the camera image, (3) moving the camera to one of two fixed Z heights, or (4) blowing a burst of compressed air into the sand directly below the camera.

We worked to reduce the size of the status image to minimize turnaround time when a command is issued. The average image size for the status image, encoded as a .gif file, is 17.3 Kbytes. Although were able to achieve response times of 10 seconds for on-campus users, cycle times of up to 60 seconds were reported from users in Europe operating via 14.4K telephone lines.

Just for fun, we created a fictional context for the system, inventing the history of a deceased paleohydrologist who had discovered unexplained artifacts in a radioactive region of southwest Nevada. We explained that the Mercury robot was originally developed to explore that region and that one mandate of our grant was to make our system "available to the broader scientific community". A hypertext document describing this background provides an online introduction.

4 Robot and Camera

The SCARA robot is an IBM SR5427 built by Smkyo in early 1980. SCARA stands for "Selective Compliance Assembly Robot Arm"; common in industrial assembly for "pick-and-place" operations because it is fast, accurate and has a large 2.5D workspace. We selected this robot over other robots in our lab due to its excellent durability, large workspace, and because it was gathering dust in our lab.

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

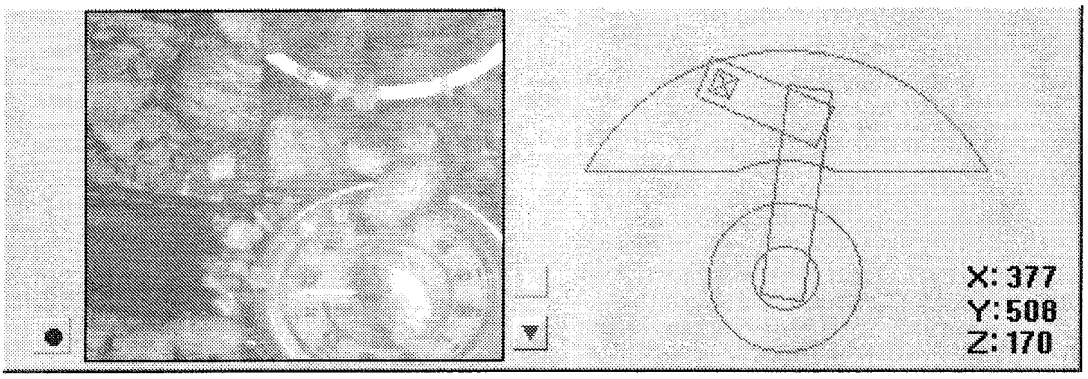

Figure 3: The "status image". At the right is a schematic top view of the semi-annular workspace and robot linkage. At left is a CCD camera image of the view directly beneath the robot end-effector. UpPown buttons are included for Z motion of the camera, and the round button is used to blow a burst of compressed air into the sand.

Unfortunately IBM no longer supports this robot and we were forced to read two antiquated BASIC programs and monitor their serial line transmissions to decipher the protocols needed for serial control of the robot. The robot accepts joint motion commands using IEEE format and checksums.

To allow users to manipulate the remote environment we initially planned to place a simple gripper at the end effector. Anticipating user attempts at sabotage (which is, after all, the time-honored hacker tradition), we opted to use compressed air as the medium for manipulation.

The CCD camera is an EDC 1000 from Electrim Inc. This camera was chosen based on size and cost. Image data is sent from the camera back through a custom serial line to a video capture card. The camera image has a resolution of 192 by 165 pixels with 256 shades of gray, which we truncate to 64 shades to reduce transfer time. Exposure time can be changed by software to range between 64ms to 200ms. Although we dowed the robot to minimize dynamic effects, mechanical settling times are long enough to cause image blur at the camera. To avoid this, we implemented a stability check by taking two images separated by 64ms and differencing them. Subsequent images are taken until the two successive images are sufficiently similar.

To avoid the complexity of another servo motor, we use a fixed focus camera and choose a focal point that compromises between the two fixed camera heights. The workspace is primarily illuminated by standard florescent fixtures. We tested a contrast enhancement routine to normalize the lighting of each image captured from the camera. This increased image quality in most cases but exaggerated intensity variations across the workspace.

5 System Architecture

As shown in Figure 4, **WWW** clients from around the world enter our system through the Internet. The system includes three communicating subsystems. Server A responds to Universal Resource Locator (URL) requests for any file on

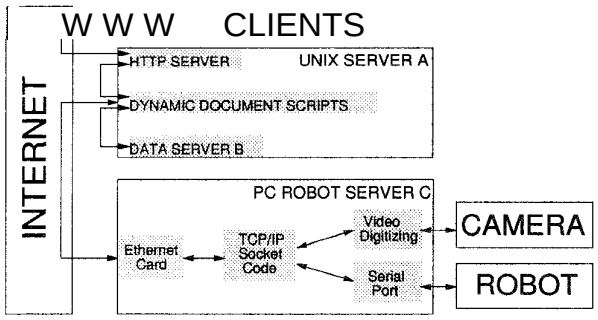

Figure 4: System Architecture

the raiders/ directory. Server A runs the vanilla NCSA HTTP Demon v.1.3 on a Sun SPARCserver 1000, with SunOS Release 5.3. Server A caches the most recent status image and sends it whenever an observer request comes in.

When a user registers as an operator by entering a password, we use a database server to verify. This server, B, runs on the same machine as Server A. The database server is custom programmed for this project, but performs fairly standard database functions.

When an operator is verified, Server A either adds the operator to the queue or communicates with Server C which controls the robot. Server A decodes the ISMAP X and Y mouse coordinates, and sends them across campus to Server C via Ethernet.

On Server C, a custom program decodes the XY coordinates into a robot command and verifies that the command is legal, e.g., within the robot workspace. If it is, the command is then executed via a command sent to the robot over a 4800 baud serial line. Once the command is completed, server C uses a local frame buffer to capture the image.

Server C then generates a new schematic view of the robot in the resulting configuration, combines it with the camera image and appropriately highlighted control buttons to form

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

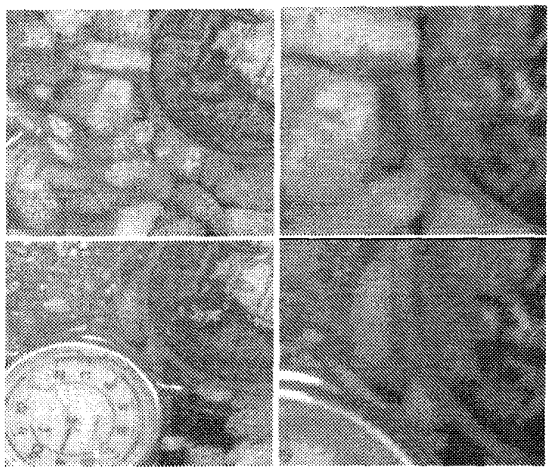

Figure 5: Sample camera images: Top row shows scene before burst of compressed air, bottom row after. Left column taken by camera in the up position, Right column by camera in the down position.

a new status image. Server C then compresses this image into GIF format and returns it to Server A, which updates the most recent status image and returns it to the client.

To maintain compatibility with the widest possible set of user platforms, we stayed within the the standard **H'ITP** protocol. For example, although X windows permit live video feed, we sacrificed this feature for the sake of compatibility. We hope that future versions of the protocol will allow the server to connect to and update clients to avoid manual re-loading of images.

The major difficulty in implementing Server C was scheduling responses to the network, the local mouse, and the image capture board. Although we seriously considered a multi-tasking environment such as Linux, the Electrim camera was only compatible with DOS and the company would not part with any source code. Thus we hand-crafted our memory management and used the screen itself as a memory buffer. This enabled us to speed a custom GIF encoder down to a few microseconds per status image.

5.1 Random Tokens

Each time Server A returns a new status image to an operator or observer, it adds a large random number to its embedded URL for the update button. This random token prevents the client from caching the status image (otherwiserepeated requests to update the image would simply reload the local image and not request an updated image from Server A).

The random token also allows Server A to identify and track clients. When an operator logs in with a verified password, Server A tracks the operator by maintaining a database of recent clients so that URL requests can be customized depending on the user's status. For example the queue is only visible to the operators and those on deck.

Servers A and B are at opposite ends of the USC campus and are connected via Ethernet. Each machine has its own IP address and resides in the usc.edu domain. Communication is achieved using a socket connection between the two machines. The implementationon Server A was doneusing the standard BSD socket functionsprovided with the SunOS4.1 operating system and Perl. On Server C we used a publicly available socket package called Waterloo TCP and Borland C. The Waterloo TCP package was obtained from the ftp site dorm.rutgers.edu in the file **/pub/msdos/wattcplwattcp.zip.**

6 Performance

We expected that the system would fail after about 6 weeks of continuous use. Although Gentner goes in to groom the sand once a day, the system is still in operation and has run unattended for the past 6 months.

Network throughput averages 20 Kbyteslsec, which is poor compared with 500 Kbytes/sec that can be achieved between two Sun workstations in close proximity on the campus network. At this time we feel that the delays are being imposed by the MS-DOS operating system running on Server C because of its inability to support networking operations and its lack of multitasking abilities, which necessitates busy waiting cycles in the PC software to obtain concurrence between the robotic/camera operations and the networking duties.

When server C detects an error, it automatically resets the robot controller, recalibrates, and returns the robot to its previous position. Also, server A automatically sends email if any of the key servers stop functioning. This occurs on average twice a month usually due to re-starts of the primary usc server. Server A also sends mail to the project team if server C stops responding, which occurs about once a month.

We monitor system usage with standard access-logs and with custom logs at Server B. In **WWW** parlance, a "hit" is a client request for a file from our system directorgr tree. In the period 1 Aug, 1994 through 1 Feb, 1995: 1,968,637 hits were made by 52,153 unique hosts (see Figures 6 and 7). If we define "uses" as clusters of hits with less than half hour idle time, the system was used 87,700 times due to repeat visits. The daily average was 430 uses which generated approximately 1000 new images. In 1994, the Mercury Project accounted for roughly half of all requests to USC's WWW server.

Space prevents us from discussing more sophisticated analysis of usage patterns such as the deterministic finite automaton transition model created by Wallace and Fisher E161.

7 Operator Logs

Some samples from over 1000 pages of operator'slogs:

From: Rex Kwok <rkwok@cs.su.oz.au>
Date: Thu Nov 3 2152:17 PST 1994

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

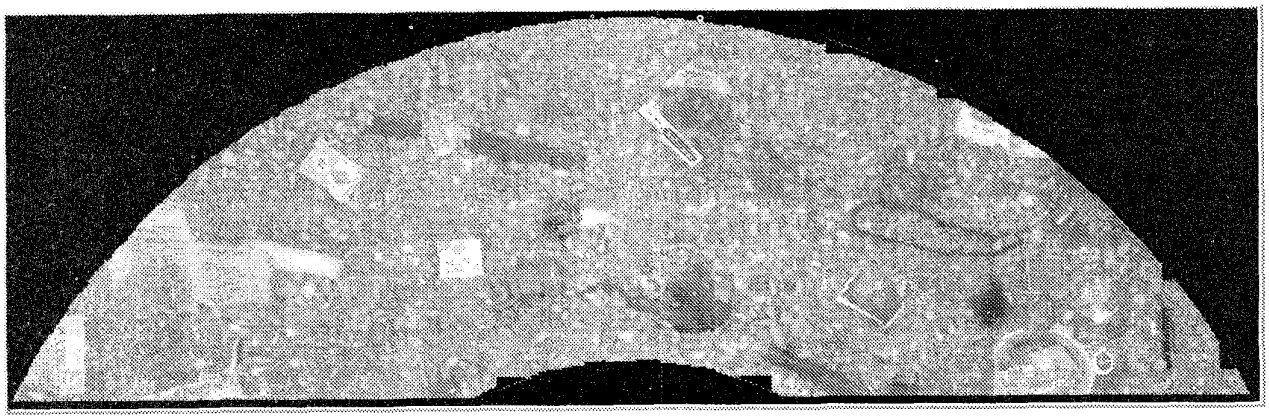

Figure 8: Composite image of workspace with artifacts such as miniature lantern, seed packet, etc..

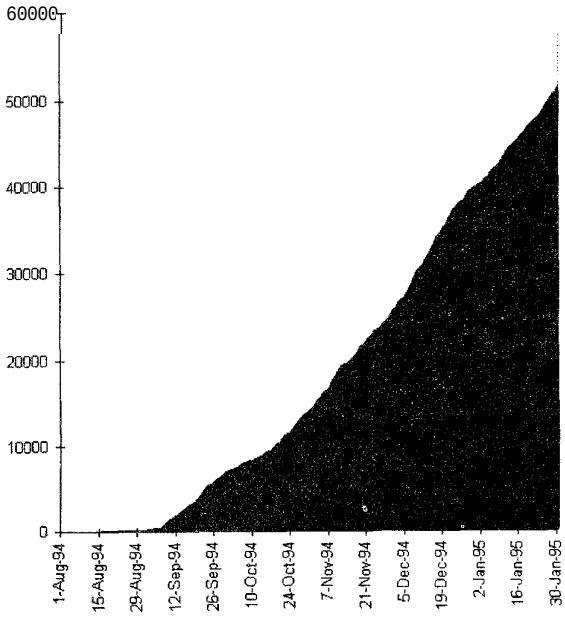

Figure *6:* Cumulative number of unique (new) hosts accessing the project.

"FANTASTIC! It is amazing to operate a robot arm from Australia."

From: Scott **Hankin <hankin@osf.org>**
Date: Fri Sep 23 09 34 59 PDT 1994:

"...this site seem; similar to the Internet. The search is analogous to trying to find something on the net, where you scan to locate areas of interest. Sometimes you'll encounter a useful nugget of information like [the antique lantern]; other times you'll discover information which seems valid but may be misleading, like the sample of 'fool's gold". Some i~formations in diferent languages, like the scrap of paper with text in English and German which points to the multinational nature of the net."

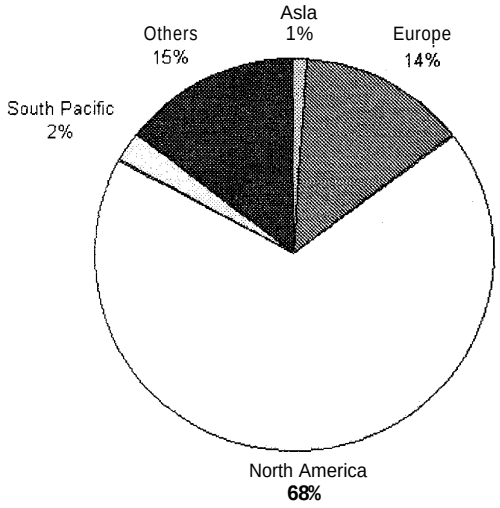

Figure 7: Breakdown of total number of hits by continent.

From: Dr. Steve M. Potter **<spotter@gg.caltech.edu>**
Date: Thu Oct 27 23:30:09 PDT 1994

"What fun! Cool idea, folks. Best use of forms and clickable maps I have seen...I was wondering how I know this is not a clever laser-disk full of pictures you grabbed, with no robot, until i saw the time on the watch after blasting it. That was when my skepticism evaporated."

And our favorite...

From: James Bryant **<jcbryant@jax.jaxnet.com>**
Date: Sat Sep 10 08:54:11 PDT 1994

"I don't believe I have seen a nicer application of science, or its money on the net."

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.