The PumaPaint Project

Matthew R. Stein
Assistant Professor of Engineering
Roger Williams University
Bristol, RI 02809 USA
mstein@rwu.edu

Abstract

The PumaPaint Project is an online robot that allows World Wide Web users to create original artwork. This paper describes the PumaPaint Project at two locations: the original site at Wilkes University and the new site at Roger Williams University. Each site allows control of a PUMA robot equipped with four paintbrushes, jars of red, green, blue and yellow paint and white paper attached to a vertical easel. A Java™ interface executing within a web browser allows interactive control of the robot. This interface contains two windows showing live camera views of the work site and various controls for connecting to the robot, viewing the task status and controlling the painting task. The original site operated from June 1998 to March 2000 with approximately 25,000 unique-addressed machines downloading the interface to produce about 500 canvases. The new site opened to the public on August 2002. This paper discusses the author's experiences in operating the original site, and the motivation for and the challenges of reviving the site in its current location.

1 Introduction

The PumaPaint Project is an online robot[1] allowing any user with a Java™ compatible web browser to control a PUMA robot located at Roger Williams University, Bristol Rhode Island. The original PumaPaint Project[2] was established at Wilkes University, Wilkes-Barre PA in 1998. Other robot sites have been online for some time allowing users to control a robot for a variety of tasks. The pioneers of web robots first opened sites in 1994, when the USC Mercury Project[3] and Australia's Telerobot on the Web[4] (independently) came on-line nearly simultaneously. These robots perform a variety of tasks, some manipulating objects, others navigating mobile robots and many offering the opportunity to manipulate a camera view. With these precedent efforts, one may ask what is new about the PumaPaint site, what contribution does it make and what motivated the authors to produce it? This paper will attempt to answer these questions. In section two we will present technical details of the site's implementation and in section three observations and discussion from operating the site. Sections four and five are summary and conclusion respectively.

In developing PumaPaint we adopted the ideas of an approach to time-delay teleoperation known as Teleprogramming[†56]. Teleprogramming has been shown reduce and potentially eliminating the delay introduced by the "move-and-wait" strategy[7]. Figure 1[8] demonstrates the teleprogramming concept. Operators interact with a virtual representation of the remote site, and from that interaction commands are sent across a distance and time barrier to the robot for execution. At its theoretical limit, the completion time of the remote task will be one round trip communication delay longer than the completion time if performed locally. Deviation from this theoretical optimum will certainly occur due to a mismatch between the virtual and real environments and/or the inability of the remote robot to execute the commands. However, to the extent that the robot can perform the desired commands based on virtual interactions, the Teleprogramming approach can provide a significant improvement over the "move and wait" strategy.

The Java™ programming language was newly released during the early development phase of this project. Java™ programs are executed (interpreted) directly on the user's machine allowing real-time closure of the operator-virtual environment loop shown in Figure 1. Using Java™, the operator interface can be freely programmed within the performance and security constraints of the user's machine and the extensive features of the language itself. The choice of Java™ was essential for a teleprogramming approach to web robotics and this distinguished the PumaPaint site from the existing web robots at the time. Since this time, Java™ has been widely adopted both as a web development tool and a general purpose programming language. Implementation of the new site included upgrading the interface to Java2 and the swing package.

™ Java is a trademark of Sun Microsystems, Inc.
† The author refined and developed teleprogramming concepts during graduate work at the University of Pennsylvania but cannot claim credit for its inception. The teleprogramming concept was first formally developed in the Ph.D. dissertation of Janez Funda, under the supervision of Dr. Richard Paul.

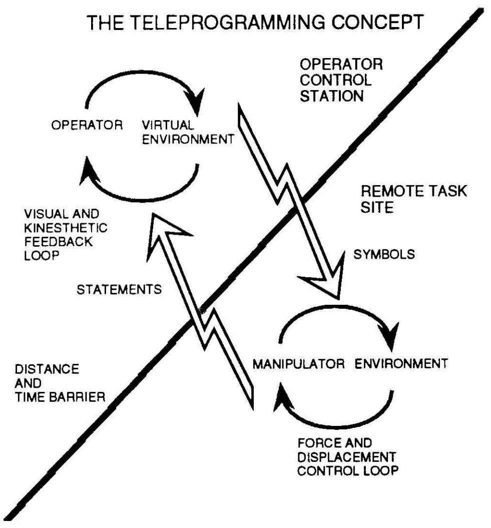
THE TELEPROGRAMMING CONCEPT
OPERATOR CONTROL STATION
OPERATOR
VIRTUAL ENVIRONMENT
REMOTE TASK SITE
VISUAL AND KINESTHETIC FEEDBACK LOOP
SYMBOLS
STATEMENTS
MANIPULATOR
ENVIRONMENT
DISTANCE AND TIME BARRIER
FORCE AND DISPLACEMENT CONTROL LOOP

Figure 1. The Teleprogramming Concept.

2. Design

2.1 Software Architecture and Design

The PumaPaint motion server was developed prior to the inception of the PumaPaint Project[9]. The motion server was designed with the concept of a daemon serving robot motions to Internet clients. The daemon waits for connections on a TCP/IP socket. When a connection is established the daemon reads and interprets fixed-size packets, each 188 bytes containing the fields shown in Table 1. This packet was kept reasonably small so that short-term storage and transmission would not be a problem, however it would be possible to make is smaller, if necessary.

Field	Size	Description
cmdflag	two byte command flag	Flag to indicate packet is command
cmd	one byte command code	One of: • t – terminate the session • r – session ready • m – move in joint mode • l – move in a straight line • g – close the gripper • o – open the grigger • p – return position information • c – transmit configuration • 0 – execute a macro
cid	four byte command number	Command number (sequential)
config	four byte character array	PUMA robot configuration string
success	two byte command flag	Flag to indicate command was successful
n	13 character string	‘n’ component of homogeneous transform
o	13 character string	‘o’ component of homogeneous transform
a	13 character string	‘a’ component of homogeneous transform
p	13 character string	‘p’ component of homogeneous transform
CmdParam1	two byte command parameter	Parameter for commands (macro number)
CmdParam2	two byte command parameter	Parameter for macro (e.g. dip depth)
passwd	11 character password	Password

Table 1. Contents of the Robot Command Packet.

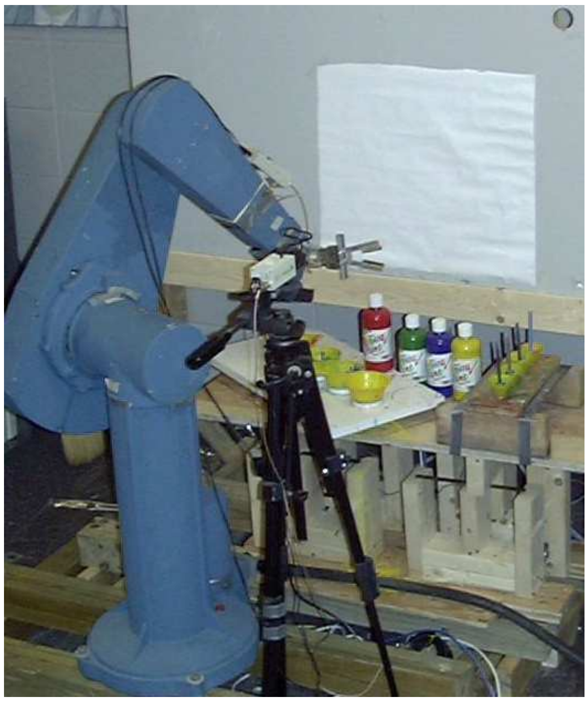

Figure 2. PumaPaint Hardware.

On receipt by the motion server, the packet is checked for validity of all the fields (for example testing all numeric vales to be within a valid range). If the packet passes these tests, it is forwarded to and immediately executed by the robot controller. When the command is completed a response packet, indicating acknowledgement and status, is returned to the sender via the TCP/IP socket. The command number (cid) field is used to monitor and display the difference between sent and acknowledged commands.

This simple set of commands implements perhaps the most trivial form of robot programming language. Packets can specify joint and Cartesian motions, gripper actions and initiate pre-programmed motions (macros). Note that no sense of time is built into the motion server and thus all moves occur at the default speed of the robot. Destinations are specified as homogeneous transforms relative to a home position, about 10cm back from the center of the easel. Macros are unconstrained subroutines compiled into the motion server. Two command parameter fields allow for a modest amount of flexibility. The first is a two-byte parameter indicating the macro number; so any number of macros can be pre-programmed. The second parameter can be passed to a macro; used, for example, to specify the depth to a pre-programmed macro to dip the brush into the paint jar.

Much of the development work on the robot motion server involved improving the tolerance to variations from normal program behavior. Because users disconnect at arbitrary times in a session, more features allowing the server to tolerate broken pipes or incomplete packets were necessary. Conversely, some proxy-servers stay connected long after the user apparently loses interest in painting, and we implemented a twenty-minute inactivity timeout to automatically disconnect machines that are not issuing commands to the robot.

Project hardware consists of a PUMA robot controlled by RCCL[10] and equipped with a parallel-fingered pneumatic gripper. The easel and all paint fixtures were constructed from inexpensive lumber and plastic, as shown in Figure 2. The paper and paint are the type frequently used by school children and the brushes are made by Crayola®. Held in fixed positions beneath plastic funnels are four paint jars containing red, green, blue and yellow, Prang™ ready-to-use tempera paint. Four plastic funnels on a small wood platform (one for each color) hold the brushes. Both the paint jars and brush holders are located on a handmade wooden platform directly beneath a homemade easel that holds a counter top in a vertically. Tonka® toy treads are glued onto the parallel-fingered gripper for better grasp stability. Both the platform and the easel are suspended on springs above contact switches so that excessive pressure will trip the switches and cause immediate power cutoff.

Despite the problems caused by inexpensive fixturing, we have intentionally declined to improve it. Devices already exist which allow the user to design graphics off-line and then produce perfect paper replicas – these are called plotters. The purpose of the site is to engage the user in the act of remote painting, and we have attempted to avoid falling into the paradigm of "tele-plotting" – the robot acting as an awkward plotter. Rather, we are attempting to reinforce the user's role as an operator who needs to observe the task and compensate for the inherent mechanical imperfections. One such imperfection is manifested in the grasping of a paintbrush handle by the parallel-fingered gripper. Variations in the grasp produce marked differences in the resulting brush stroke, and if the user seeks to control the results, he or she needs to account for this variation.

A second computer is used to run the Hypertext Transfer Protocol Daemon (httpd) and the serve the video images. A security feature of Java™ limits an applet to connecting only to the machine of its origin, so Java™ classes loaded from this machine may not directly connect to the robot motion server. Instead, an intermediate daemon accepts connections and routes packets to the robot motion server. This security feature prevents web users from directly connecting to the real-time control machine.

2.2 PumaPaint Java™ Interface

Because the Java™ applet is executed on a web user's machine, the user interacts directly with the applet and receives immediate feedback. The interface takes advantage of this feature, providing with two channels of feedback: one immediate and virtual and the other time-delayed and real as depicted in Figure 1. Figure 3 shows the latest interface, as a user would see it within a browser window.

The center portion is a virtual canvas and the main area of interaction. By clicking and dragging the mouse in this area the user issues commands to the remote robot. These mouse actions also cause the selected color to appear on the virtual canvas. We feel, however, that simply tuning virtual canvas pixels beneath the mouse to a selected color would mislead rather than assist the user, so several features are added to attempt to increase the fidelity of the virtual canvas. The virtual canvas is colored as a blob, rather than a shape with sharply defined edges. The blobs contain randomly generated gaps and streaks, and the proportion of area turned to the selected color progressively decreases as the brush stroke continues. This simulates the effect of depleted paint in an attempt to remind the user to manually replenish the paintbrush. Another aid simulates colors mixing on the canvas. Should a red brush stroke be made over a yellow stroke on the virtual canvas, the resulting overlap will appear orange.

The left panel allows the user to set two parameters determining how the paint will be applied to the canvas. The "Dip" slider controls the depth of the dipping motion into the paint jar and thus the quantity of paint on the brush. The "Pressure" slider, named to give the lay user an intuitive notion of its function, only specifies the distance the brush will travel toward the easel and not the pressure exerted. Setting the "Pressure" to a high value (labeled "Squash") will usually cause a hard contact between the brush and the easel, while at a low value (labeled "Hover") the brush tip will not contact the easel. The panel contains buttons for the four colors of paint and the "dip" button. Once a color is selected the dip button turns the selected color as a reminder to the user what color will be dipped.

The top row has "Cam" buttons that will spawn a new window containing an automatically updated image from the live video cameras. Video update rates depend on the quality of the image selected via a slider in the camera window. The maximum update rate approaches live video rates but typical update rates will be slower depending on image size and transfer time.

The right panel shows command number, success flag and current position. Despite the typical user's lack of interest in this area, it shows the difference between packets sent and received and the packet queue. We believe these fields are essential to understanding the issues of time-delayed teleoperation, so we include them in the hopes that a teachable moment will arrive. The "Diff" field shows the difference, in command number, between the last command sent and the last acknowledgement received. This field will show that the robot is almost always behind the user, and sometimes quite far behind. If the user wishes to "wait for the robot", he or she is waiting for the robot to process every command sent indicated by a zero in this field. The packet queue shows the length of the queue of commands waiting to be sent to the robot. Irrespective of communication delay, it takes much less time for the user to specify commands (via mouse clicks) than it does for the robot to perform these commands. For example, it takes virtually no time for the user to click the "dip" button, but about twenty seconds for the robot to perform this action. Thus, in a matter of minutes it is easily possible for the user to specify hours of robot motion. This is not likely to be useful, so the length of the outgoing packet queue is limited to two hundred commands, equivalent to about ten minutes of robot motions depending on what the commands are. If the queue reaches

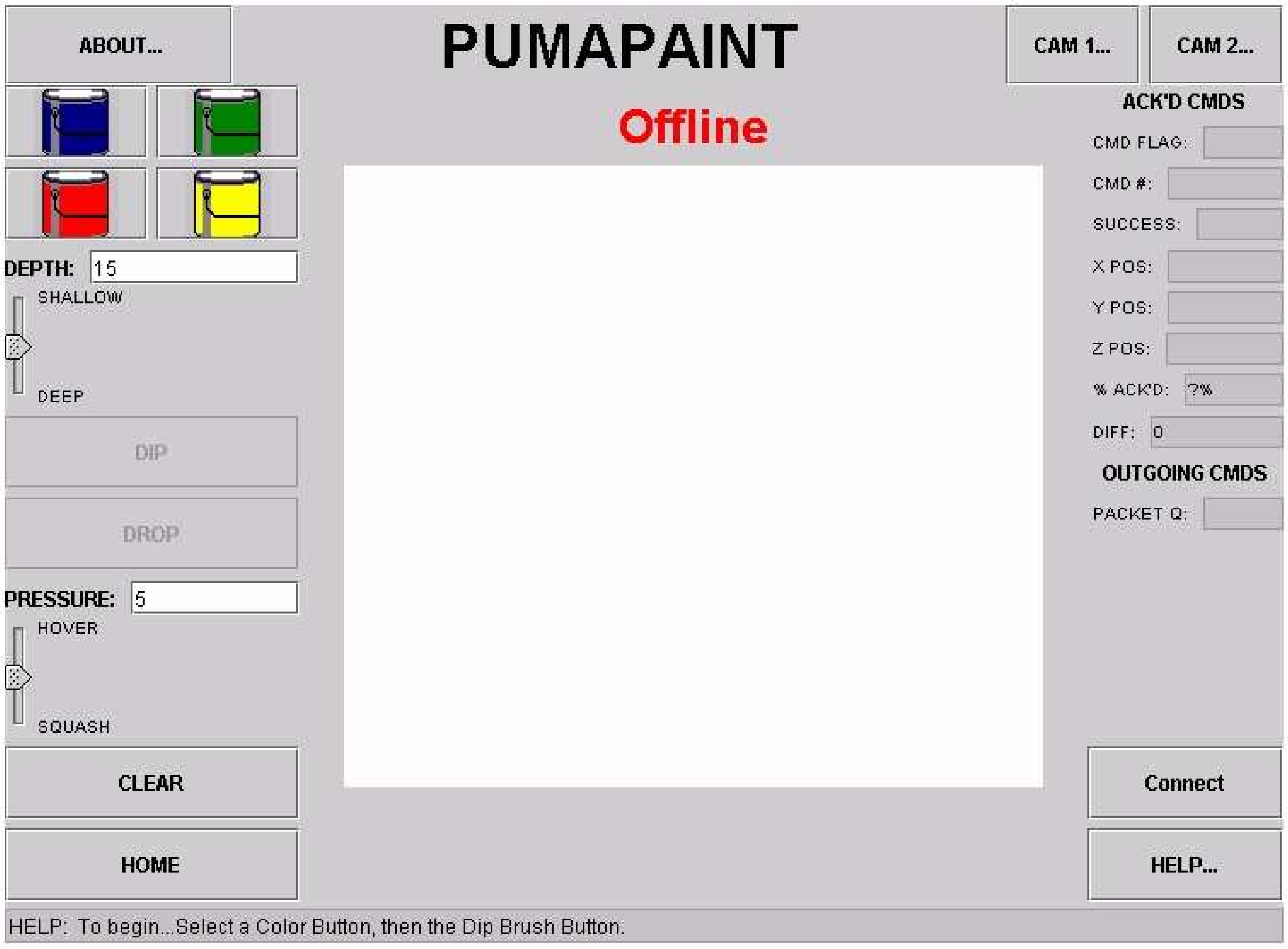

Figure 3. The PumaPaint Interface

this length, the interface notifies the user that he or she is "too far ahead" of the robot and has the option of either waiting or flushing the outgoing command queue.

Each of these features is an attempt to aid the user unobtrusively. For example, virtual brush stroke may become clear to remind the user to replenish the paint, but no attempt is made to take this control away from the user and replenish automatically. The "Pressure" can be set between numeric values labeled as "Hover" and "Squash". "Squash" will do just that, compressing the brush against the easel. Although this often results in ugly splotches of paint on the canvas and wear and tear on the brushes, we do not prevent the user from doing this.

Painting is most likely to be successful if the user understands that the real canvas will **never** exactly duplicate the virtual canvas. The only certain means of determining the result of commands is to view the easel using the two provided camera images. One camera is situated to view the entire canvas from a slightly oblique angle, approximately matching the virtual canvas. Another camera is mounted on the robot to provide a close up view of the brush contacting the canvas.

The new Java™ interface was developed using the Sun Microsystems Java(TM) 2 SDK, Standard Edition, v 1.3.1_04 for Linux, using the v 1.3.1API Specification and the javax.swing package. As of this writing, it seems many browsers are not compatible with the Java2/Swing and this requires the user to download the Java2 Runtime Environment (JRE). We discovered this after the Java2 interface was developed, so we currently find we are unwittingly ahead of the curve with respect to browser support.

3. Site Experiences

Daily maintenance of the PumaPaint site is neither challenging nor particularly time-consuming. Software maintenance involves only capturing images of the current canvas and publishing it to the web page. The author places a fresh sheet of paper on the easel, cleans the brushes and fills the paint jars before leaving each evening. The daily activity is cleaning up the mess from dropped brushes and changing the paper. The deliberate weak-link of the mechanical system is the rubber pads hot-glued to the parallel-finger gripper, and these typically tear off during crashes. Normally all that was required was to re-heat the glue and stick these back on.

3.1 Interface statistics

We analyzed the access log of the http daemon for the first year of operation using the program *Analog*[1]. Table 2 shows the number of downloads for the main PumaPaint HTML web page; for the Java™ interface; and for the communications class. The later is only loaded after the interface is operational; the user has entered a password and then selected a "Connect to Server" button. Although this counts users gaining control of the robot, it still does not indicate if the user immediately disconnected or actually applied paint to the canvas. Distinct hosts downloading the communication class is the most conservative statistic and is probably the truest measure of the number of persons actually using PumaPaint in the first year.

Hosts downloading PumaPaint components	**Total**	**Distinct**
Downloads of the main PumaPaint HTML page	18,039	11,813
Downloads of the interface	9,802	6,648
Downloads of the communications class	6,927	5,593
Average successful requests per day	18	

Table 2. PumaPaint download statistics for June 3rd 1998 to June 3rd 1999.

Table 2 indicates there is about a 3000 (or almost 30%) difference between total downloads of the interface and total downloads of the communication class. It is difficult to ascertain from the access log the source of this difference but one could conjecture the following causes:

- **The user is just "wandering through".** A user follows the link for the interface not genuinely understanding the site or the meaning of the link and then immediately changes his or her mind and presses the "Back" button
- **The user tires of waiting.** The first real delay using the PumaPaint site occurs when the interface is downloaded. The user's machine must start the Java interpreter and then load the classes necessary to execute the interface. Users may begin to download the interface intending to use it, but then lose interest in the face of the download delay.
- **The user gets confused or confuses the interface.** During the download delay the user may click the mouse and/or raise and lower windows. The password window is a separate window that can be easily lost or closed in this process. If this happens the user will eventually see the entire interface but it will be non-operational. The user would, undoubtedly, get frustrated with the non-working interface and exit.
- **The browser is not capable of interpreting Java.** If the user's browser is not equipped with a Java interpreter (more recently the java2 interpreter) and thus following the interface link will produce a blank screen in the browser. There is currently a note explaining where to get the JRE, but users may not bother with that process.
- **The interface simply doesn't work.** This should not be overlooked as a potential cause of users not connecting to the robot, nor should the mere fact that numerous people have used the interface indicate that all users could. Because of the diversity of browsers, platforms and Java implementations available, I cannot be certain that the interface applet will actually work in all cases. Although I have no direct evidence of this, I suspect, from my own experience with Java, that the interface is simply failing for a significant number of users.

One statistic (noted in Table 2) is the average successful downloads of the communications class per day is 18, or less than one per hour. As most users spend a short time actually in control of the robot, this statistic should demonstrate this author's observation: the robot sits idle most of the time. The site has only seen continuous use (one user right after another) for a few brief periods coincident with prominent press coverage. The majority of the time anyone in the world interested in doing so may take control of the robot with no waiting.

3.2 Some notes about camera images

Records of the image downloads also provide some insights into site usage as well as user preferences and behavior. During a typical painting session a user will view ten to one hundred images, so the number of image downloads far exceeds the

[1] *Analog* is copyright (C) Stephen Turner and is freeware available at http://www.analog.cx/

number of users. The site has two cameras; Camera 1 is mounted on the arm showing a close-up view of the gripper holding the paintbrush (see Figure 4). Camera 2 is a "canvas shot" in which the user views the entire canvas and frequently the robot (see Figure 7A).

Image file downloads from July 10th, 1998 to June 3rd, 1999	**Camera 1**	**Camera 2**
Total number of downloads	153540	168041
Downloads at default setting of 10 (%)	116007 (75.5%)	120572 (71.8%)
Downloads at lower quality than default	10197 (6.6%)	13543 (8%)
Downloads at higher quality than default	27336 (17.9%)	33926 (20.2%)

Table 3. Image download statistics for July 10th 1998 to June 3rd 1999.

Table 3 shows some of the statistics on camera image download from July 10th, 1998 to June 3rd, 1999. The default setting is 10 on a scale of 1 to 100, and this produces a 12KB image file that is noticeably blotchy. I felt that the blotchy image would be a visual clue that it is possible to modify the quality with the slider.

While about 70% of all users either never changed the default setting (or set it back) the remainder did change the default setting with the majority opting for a higher quality (and typically slower) image. Figures 5 and 6 show a histogram of the JPEG image quality for all downloads for camera 1 and camera 2. Because the number of downloads at the default setting (10) is over 10 times the next highest setting, the scale is adjusted to highlight downloads at the non-default settings in histogram bins of 5. Note that these figures represent the total number of image downloads and can only roughly be correlated to the number of users choosing that quality setting. Figures 5and 6 show users adjusting the quality setting had a tendency to shift quality to the lower end of the scale and had little interest in images with a quality setting between fifty and one hundred. Lower quality images are smaller (typically loading faster) and users were actively trading off between image quality and download speed. One interesting statistic is that 577 different users downloaded 6863 images with a quality setting of 1. As it can be seen from Figure 7B these images are very coarse and, in some cases, the image may be unrecognizable. Setting the image quality to 1 reduces the image to 6Kb and apparently a substantial number of users preferred the faster download time despite the rough image.

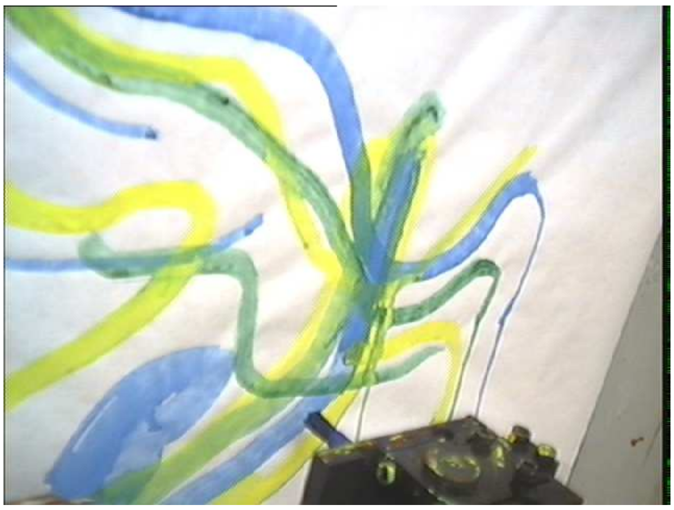
Figure 4 Camera 1 view of the easel, the robot end effector, and the tip of the paintbrush. (Wilkes site)

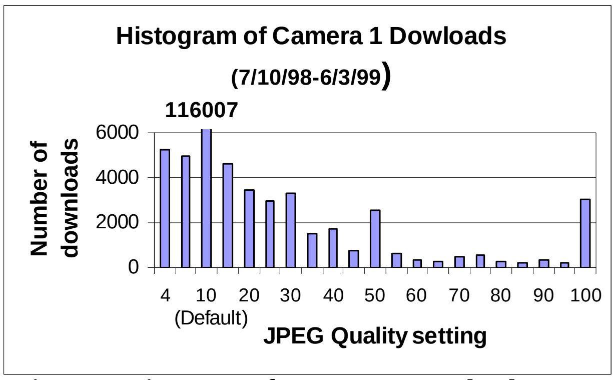

Figure 5 Histogram of Camera 1 Downloads

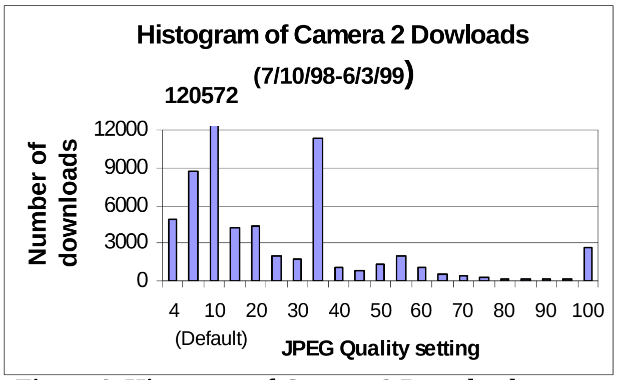

Figure 6 Histogram of Camera 2 Downloads

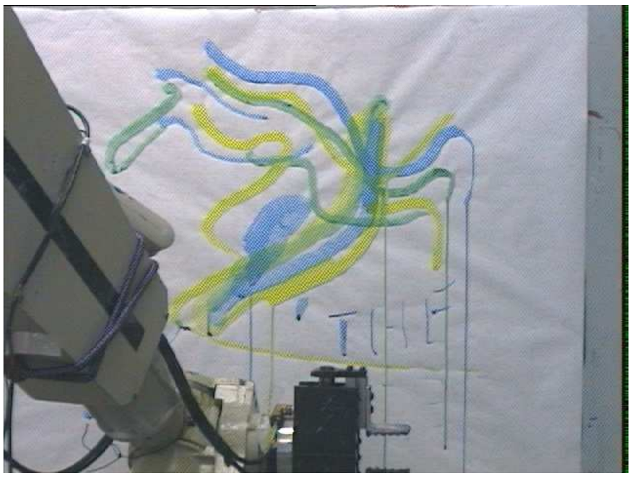

Figure 7A Camera 2 at quality setting 50 (Wilkes site)

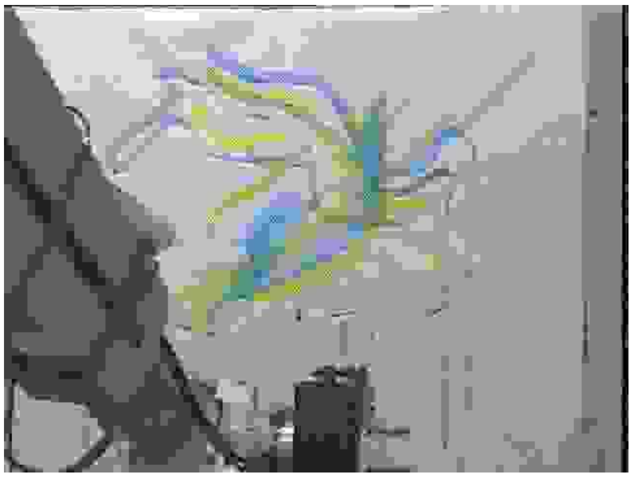

Figure 7B Camera 2 at quality setting 1 (Wilkes site)

3.3 A look at the paintings

Perhaps the most unique aspect of the PumaPaint project is the spontaneous creation of physical artifacts via the Internet. Given this unprecedented[†] capability, perhaps the most interesting question is: “What did they do with it?” PumaPaint users produced 500 painted images in the first year of operation, and examination of these images provides, to the best degree possible, the answer to this question. In summary, the canvases can be divided into five major categories: **nothing, text, icons, vandalism, and art**.

Of the 500 canvases, 60% contain test strokes, spurious lines, squiggles or other marks not identifiable as an attempt to create a meaningful image. This is often the result of leaving the canvas in place and allowing multiple users to scribble on it. Figure 8 shows a canvas created April 6th, 1999. Note that test strokes from subsequent users obscure the strokes of a previous user. Paint drips, small geometrics and text appear on many canvases.

Some users sought to vandalize the canvas, and to do so was entirely possible by setting the pressure to “squash” and repeatedly moving over the same area. The apparent goal was to rip off the paper and paint on the easel, as shown in Figure 9. It is somewhat difficult to count deliberate acts of vandalism because the robot has enough mishaps on its own. The “blooper reel” portion of the PumaPaint site shows 30 different incidents of the brushes or paper being mishandled. The robot is entirely capable of doing this without user intention and there was conclusive evidence of vandalism in only about five cases. Thus, although vandalism was an option that some users chose, it was far from common or prevalent.

[†] To the Author’s knowledge, PumaPaint is the first site that allows the general public to create unique, personally identifiable artifacts on the World Wide Web and receive them in the mail.

The “Hall of Fame” section of the site contains 50 images that the author considers fine works of art or deserving of special attention and Figure 10 is a collage of some of these images. Although land/seascapes are often attempted, the author found the greatest appeal in portraits.

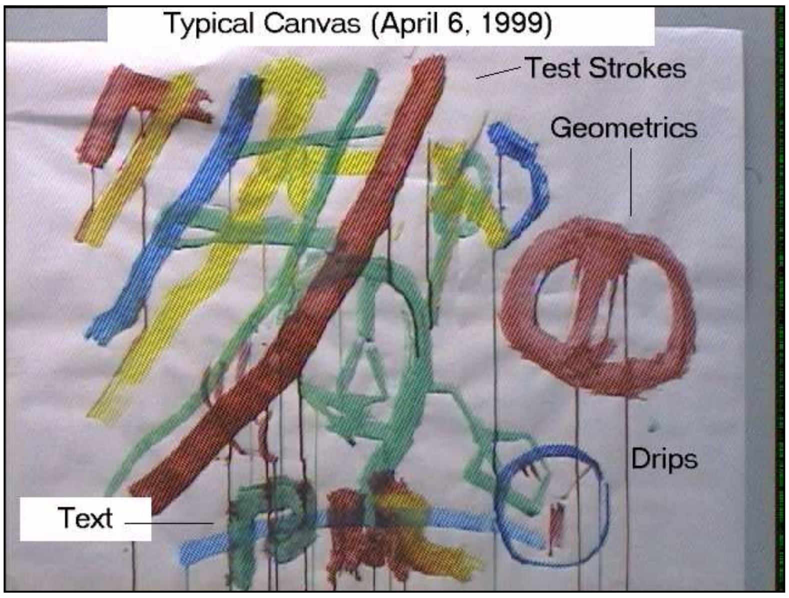

Figure 8. Examination of a Typical Canvas.

4. Summary

Despite the precedent efforts in online robots, the author expected that this site would also be of interest because of the chosen task of painting and the ability to physically receive the artwork. Users were inherently interested and accustomed to the task of painting as well as the notion of putting one stroke at a time on a canvas. Although some results were anticipated by the author, unanticipated results may be the more interesting and informative. The outcomes listed below went directly counter to the author’s expectation or were completely unanticipated.

- **The lack of commitment demonstrated by most users**

The offer to mail a user a canvas free of charge is prominently featured on the web page, but few users took advantage of this offer. As a result, the author has accumulated over 500 canvases waiting for someone to claim them. The majority of users gain control for a short while, make a few trial strokes and depart without comment.

Most surprisingly, creators of half of the images in the “Hall of Fame” never self-identified or requested their images. Some of the most elaborate images took 30-60 minutes to create, yet the artist never sought to claim them. The author can only conjecture that the artists were either unaware of the offer, felt there must be some catch or otherwise feared identifying themselves.

- **Infrequent vulgarity and vandal/hacker attacks**

The author was completely surprised by the apparent absence of hacker/cracker attacks on the site. We anticipated that there would be an strong appeal to potential hackers to wreak actual physical havoc with a robot connected to the web, and we spent a good deal of time installing physical protections, proxy servers and password systems. There has been no known attempt to compromise these precautions. To vandalize the site only requires moving the pressure setting to “squash” and moving over the same area until the paper tears. Although up to thirty users chose to do this, this number is much smaller than the author’s expectation.

One quite unexpected outcome is of the over 300 canvases containing text, less than 10 contain text that is clearly identifiable as profanity or vulgarity. For the most part, the canvases resemble graffiti, yet most users chose to leave positive messages or a simple self-identification. This suggests that the public World Wide Web space created by the PumaPaint site is somewhat different than physical public space.

- **The lack of a serious artist or request for better equipment**

The use of inexpensive paints and paper results in most paintings frankly looking like elementary school work suitable for hanging on the refrigerator. Undoubtedly, finer brushes, paper and paint would result in much more beautiful images, but as of yet, not a single user has requested this. Even of the three known users who frequented the site and produced images of increasing quality, none requested better equipment.

- **The lack of interest in images**

Despite a prominent message on the main page begging users to look at the camera images before painting, a great many users never downloaded a camera image. In developing this site, the author anticipated that witnessing the robot perform commanded motions in real life would be the main fascination with the site, but a good portion of users had no such interest or did not understand that this was possible. Only a small set of users viewed the video feedback continuously and made corrections to the canvas in an attempted to create artwork.

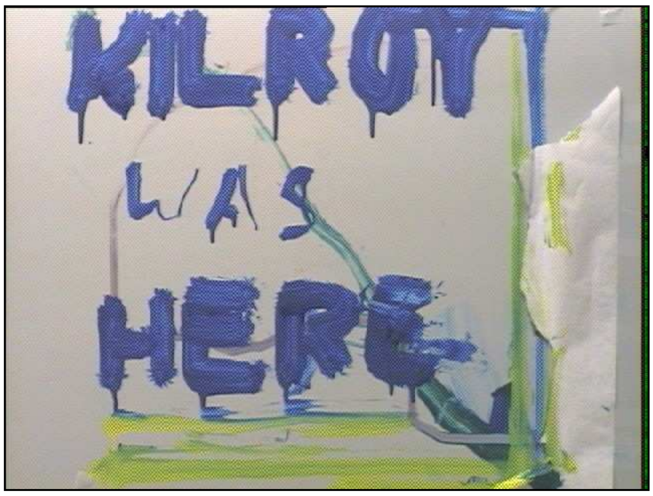

Figure 9 Example of Vandalism.

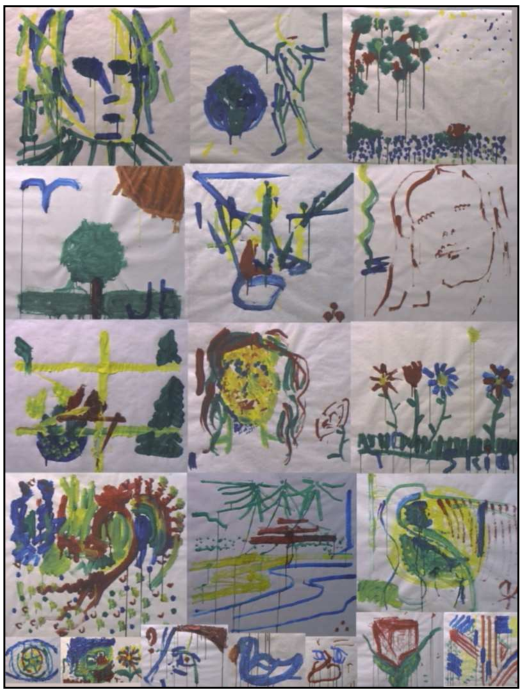

Figure 10 Collage of finest images.

5. Conclusion

5.1 Validation of Teleprogramming.

However modest is the achievement of creating artwork remotely, it does demonstrate the fundamental concepts of teleprogramming. Whether the artwork is good or bad, it is without a doubt the result of intentional control of a distant operator via the Internet. In addition, the PumaPaint interface allowed the operator to create masterpieces (or junk) without adopting a visual-feedback based move-and-wait strategy. Operators interacted with a virtual interface and received immediate virtual feedback while this interaction caused a series of commands to be generated and later executed by a remote robot.

As anticipated for a teleprogramming system, we repeatedly observed the robot performing commands far, often minutes, behind the operator's action, even without communication time delay. This de-synchronization between the operator's commands and robot actions is the characteristic that distinguishes a teleprogramming approach to remote robotics. The PumaPaint site is a partial validation of the Teleprogramming concept because it demonstrates operators in control of a task despite de-synchronization and in the presence of limited bandwidth and time-delayed communications.

We do not wish to overstate its significance of the PumaPaint site and therefore also note limitations and simplifications of this demonstration, enumerated below:

- **No error correction**. No facility is provided to recover from execution errors, yet this is perhaps the most significant challenge to time-delayed telerobotics[11]. Autonomous operations, such as dipping and brush handling frequently encounter errors, but the user has no ability whatsoever to address or correct these.
- **Easy mechanical tasks**. The mechanical interaction between brush and canvas is deliberately compliant and imprecise. As a result, users do not have a need or interest in carefully controlling this interaction.
- **Inherently imprecise task**. Operators had imprecise expectations as to what will appear on the canvas and were tolerant of variations. Thus, although operators were able to perform the task independent of time delay, they were also not too concerned with how it turned out.
- **No fear of damage.** The fear of damage that requires a remote operator to make guarded moves or adopt a move-and-wait strategy is lacking from this demonstration.
- **Easy to model.** Application of paint to a canvas was the only germane interaction between the robot and its environment and this could be modeled with relative ease. The limitation of requiring a high fidelity model to simulate all significant components of the robot-environment interaction is lacking from this demonstration.

A driving interest in the project for the author is curiosity as to what the world at large would do with such a capability and what artwork could be produced. After years of operation we have in hand a body of evidence as to the answer to these questions. As to what will be created – the predominant answer is **nothing**. Not particularly worthwhile or interesting images are the majority of canvases produced. As to the fascination the world has in such a project – the answer is predominantly: passing interest. Most people find the idea mildly interesting and are willing to try it out, but few take the time to explore the possibilities in any depth.

5. 2 What's Next

PumaPaint site has been revived at Roger Williams University for the purposes of finding out what, if anything can be done to improve it. The new site has much of the same elements of the original Wilkes site: a PUMA robot, four colored paintbrushes and two cameras and a paper easel. Only the video hardware is significantly improved over the original site.

Computer Science students will be experimenting with various improvements to the interface. One modification currently underway is closer integration of the real camera image with the virtual canvas. There are some subtle issues to be worked out, for example, should the camera image overwrite the virtual canvas completely, or should users somehow be able to view both? Locations on the camera image will not match locations on the virtual canvas unless the camera is carefully calibrated. Can we devise an interface allowing the user to adjust the mapping between the camera image and the virtual canvas? One factor determining the success of these interface improvements will be seeing if the art improves.

6. References

[1] http://pumapaint.rwu.edu
[2] http://yugo.mme.wilkes.edu/~villanov
[3] K. Goldberg, M. Mascha, S. Getner, N. Rothenberg, "Desktop Teleoperation via the World Wide Web" 1995 IEEE International Conference on Robotics and Automation, Nagoya, JAPAN
[4] K. Taylor, J. Trevelyan, "Australia's Telerobot On The Web", *26th International Symposium On Industrial Robots*, Singapore, Oct. 1995.
[5] R P. Paul and C. P. Sayers and M. R. Stein. "The Theory of Teleprogramming", *Journal of the Robotics Society of Japan, Vol. 11, No. 6 1993*.
[6] J. Funda, "Teleprogramming: Towards Delay-Invariant Remote Manipulation", *Ph.D. Thesis, University of Pennsylvania, 1992*.
[7] . William R. Ferrell and Thomas B. Sheridan, "Supervisory Control of Remote Manipulation", IEEE Spectrum Vol. 4 No. 10 1967.
[8] M.R. Stein, "Behavior-Based Control For Time Delayed Teleoperation", *Ph.D. Thesis, University of Pennsylvania MS-CIS-94-43*.
[9] M.R. Stein, C. Ratchford, K. Sutherland and D. Robaczewski. "Internet Robotics: An Educational Experiment". Telemanipulator and Telepresence Technologies II. SPIE Volume 2590 1995.
[10] J. Lloyd, V Hayward. "Real-time trajectory generation in Multi-RCCL", *Journal of Robotic Systems*, vol. 10, no. 3, p.369-90
[11] M. R. Stein, C. P. Sayers and R. P. Paul, "The Recovery from Task Execution Errors during Time-Delayed Teleoperation". *International Conference on Intelligent Robots and Systems - IROS '94*, Munchen, Germany, 1994.