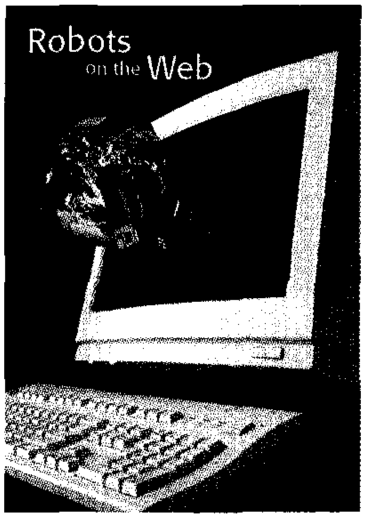


Lessons Learned from Xavier

Ongoing Experiments in Interactive, Web-Based Robotics with an Autonomous Mobile Robot on the Web

For the past four years, we have been running an experiment in web-based interaction with an autonomous indoor mobile robot. The robot, called Xavier, can accept commands to travel to different offices in our building, broadcasting camera images as it travels. The experiment, which was originally designed to test a new navigation algorithm, has proven very successful, with nearly 40,000 requests received and 240 kilometers traveled to date. This article describes the autonomous robot system, the web-based interfaces, and how they communicate with the robot. It highlights lessons learned during this experiment in web-based robotics and includes recommendations for putting future mobile robots on the web.

by REID SIMMONS, JOAQUIN L. FERNANDEZ, RICHARD GOODWIN, SVEN KOENIG, and JOSEPH O'SULLIVAN

The Creation of Xavier

In December 1995, we began what we assumed would be a short (two- to three-month) experiment to demonstrate the reliability of a new algorithm that we had developed for autonomous indoor navigation [12]. To provide a continual source of commands to the robot, we set up a web page in which users throughout the world could view the robot's progress and command its behavior. What we failed to anticipate was the degree of interest an autonomous mobile robot on the web would have. Even now, years later, Xavier continues to fascinate people who have read about it in the popular press, stumbled across its web page, or found it through one of the many links to its website (http://www.cs.cmu.edu/~Xavier).

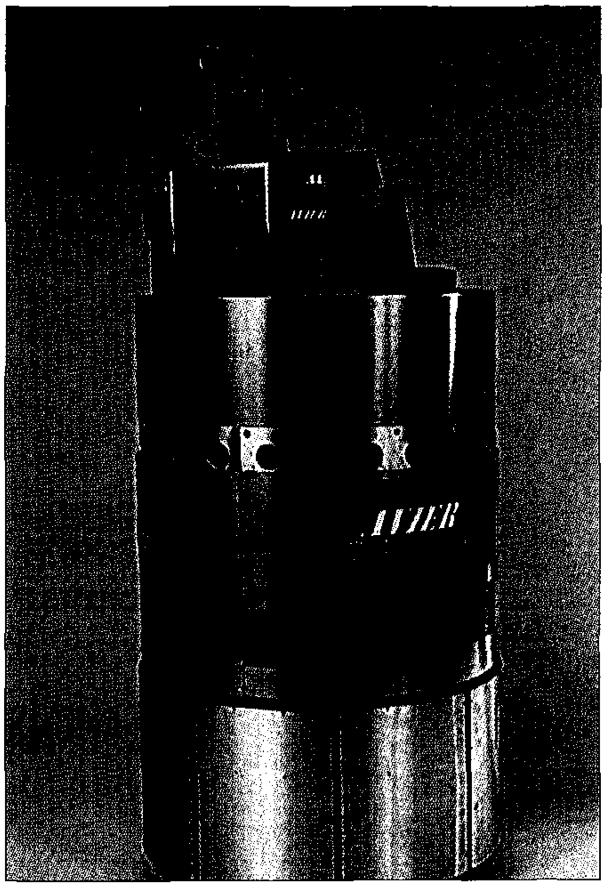


Figure 1. *The Xavier Robot.*

Xavier (Fig. 1) is built on top of a 24-inch diameter base from Real World Interface. The commercial base is a four-wheeled synchro-drive mechanism that allows for independent control of the translational and rotational velocities. The torso and superstructure of Xavier were designed and built by a class of computer science graduate students in 1993. The sensors on Xavier include bump panels, wheel encoders, a 24-element sonar ring, a Nomadics front-pointing laser light striper with a 30-degree field of view, and a Sony color camera on a Directed Perception pan-tilt head. Xavier also has a speaker and a speech-to-text card. Control, perception, and planning are carried out on two 200 MHz Pentium computers, running Linux. A 486 laptop, also running Linux, sits on top of the robot and provides for graphical display and communication to the outside world via a Wavelan wireless Ethernet system. The three onboard computers are connected to each other via thin-wire Ethernet.

In general, autonomous mobile robots do not provide the same type of immediate feedback as do teleoperated robots.

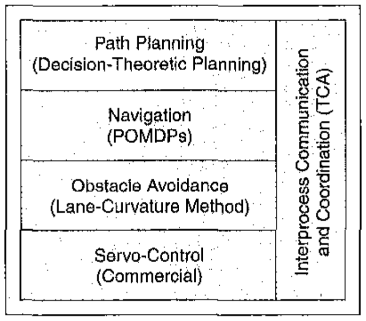
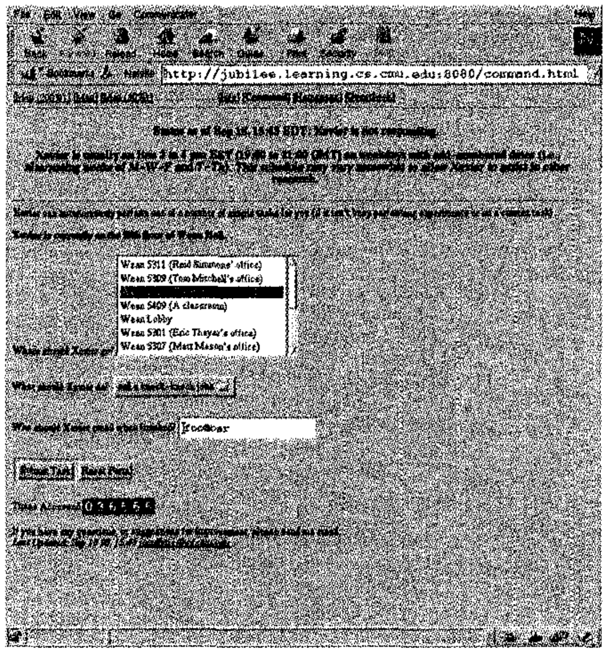


Figure 2. *Xavier's navigation system.*


File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop

http://jubilee.learning.cs.cmu.edu:8080/command.html

Wean 5311 (Reid Simmons' office)
Wean 5309 (Tom Mitchell's office)
Wean 5409 (A classroom)
Wean Lobby
Wean 5301 (Eric Thayer's office)
Wean 5307 (Matt Mason's office)

foo@bar

Figure 3. *Command interface web page.*

Although our main research focus has not been the web-based aspects of Xavier, the experiment has taught us several things about the interactions between remote users and autonomous robots. The main lessons learned concern robot reliability and the types of remote interactions that are useful, together with some sociological anecdotes about people, technology, and the web.

Xavier differs from most other web-based robots in that it is mobile and autonomous (the Rhino and Minerva tour-guide robots are other highly successful web-based autonomous mobile robots). Mobility impacts web-based robots because the bandwidth achievable by (affordable) radio modems is rather limited. Thus, real-time visual feedback and control is often difficult to achieve, especially if the workspace of the robot is a large area (such as a whole building) so that radio coverage becomes a factor. Also, battery power is limited, so the robot can operate only a few hours per day.

Autonomy can help in reducing the bandwidth requirements for control, but this introduces problems of its own, particularly in the area of interactivity. People seem to prefer "hands on" control—in general, autonomous mobile robots do not provide the same type of immediate feedback as do teleoperated robots. This is exacerbated by the limited up-time of the robot, which reduces the chances for people to see the robot actually operating when they happen to come to its website.

Despite these challenges, we believe that Xavier has been quite a successful (if somewhat inadvertent) experiment in web-based robotics. In particular, it has given thousands of people their first introduction to the world of mobile robots. Judging by the feedback we have received, the overall response to Xavier has been extremely positive, and people are generally very impressed that robots can have such capabilities (many want one for their own).

Autonomous Navigation System

The Xavier navigation system is a layered architecture (Fig. 2), consisting of servo-control (provided by the commercial base and pan-tilt head), obstacle avoidance, navigation, and path planning. Each layer receives "guidance" from the layer above and provides commands to the layer below. Each layer also filters and abstracts information for the higher layers, enabling them to operate more globally without getting swamped by data. While the navigation system and each of the individual layers have been described elsewhere [11], we give a brief overview here of the salient features of the system.

The **servo-control** layer, which controls both the base and pan-tilt head, provides simple velocity and/or position control. It also provides feedback on command execution and position information, based on encoder readings. The servo-control layer is primarily commercial software that comes with the hardware.

The **obstacle avoidance** layer keeps the robot moving in a desired direction, while avoiding static and dynamic obstacles (such as tables, trash cans, and people). It uses the Lane-Curvature Method [8], which tries to find highly traversable lanes in the desired direction, and it uses the Curvature-Velocity Method [10] to switch between lanes and avoid dynamic obstacles. Both methods take vehicle dynamics into account to provide safe, high-speed motion (Xavier averages about 45 cm/sec in peopled environments).

The **navigation** layer is responsible for getting the robot from one location to another. It uses a partially observable Markov decision process (POMDP) model to maintain a probability distribution of where the robot is at all times, choosing actions based on that distribution [5, 7, 12]. Thus, while the robot usually never knows precisely where it is, it rarely gets lost.

Currently, the tasks that Xavier can perform at a destination include taking a picture, saying "hello," and telling a robot-related knock-knock joke.

The **path-planning** layer determines efficient routes based on both a topological map that is augmented with rough metric information and the capabilities of the robot. It uses a decision-theoretic approach to choose plans with high expected utility, taking sensor and actuator uncertainty into account [5]. For instance, if there is a reasonable chance that the robot will miss seeing a corridor intersection (and thus have to backtrack), the planner might choose a somewhat longer path that avoids that intersection altogether.

The Xavier navigation system is implemented as a collection of asynchronous processes, distributed over the three computers on board Xavier. The processes are integrated and coordinated using the Task Control Architecture (TCA). TCA provides facilities for interprocess communication (message passing), task decomposition, task synchronization, execution monitoring, exception handling, and resource management [9]. Using TCA, new processes can be easily added and removed from the system, even as it is running.

In addition to the layers described above, there are processes that control the camera and pan-tilt head, provide speech generation, and monitor the robot's execution and recover from failures [2].

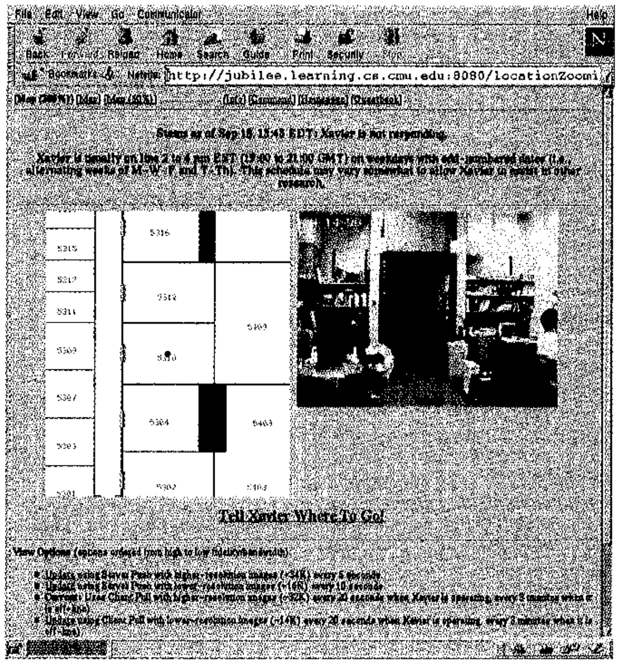
Web-Based Interface

The World Wide Web interface was designed with the intention of making it easy for nonroboticists to interact with the robot. The command interface web page (Fig. 3) shows Xavier's current status (updated every 5-10 seconds). It provides a discrete list of destinations to send the robot (about a dozen different locations, mainly offices and classrooms, for each floor of our building), a list of simple tasks to perform at that location, and space to enter an (optional) e-mail address. Currently, the tasks that Xavier can perform at a destination include taking a picture, saying "hello," and telling a robot-related knock-knock joke (the overwhelmingly favorite task).

When a user submits a task request, a confirmation web page is sent back that indicates when the robot will likely carry out the task (either immediately, if the robot is operational and not busy; at some time in the near future, if it is up and busy; or some time in the indefinite future, if the robot is not currently on line). If the request includes a legitimate e-mail address, Xavier will send e-mail after it achieves the task, and it will include a mime-encoded image (gif format) showing what it saw when it reached that destination (plus the text of the knock-knock joke it told, if that was its task).

In addition to the command interface page, there is a monitoring web page that includes the robot's current status, a zoomable map of the floor Xavier is currently on, and a color picture of what it currently sees (Fig. 4). Both the map and the camera image are sent as gifs and are updated every 5-10 seconds. The map shows the area around the robot and its most likely pose, based on the probability distribution the robot maintains. Additional web pages include information about our lab, statistics on Xavier's performance, a guestbook, and a "robot joke contest" page.

The web-based interface is implemented as one additional onboard layer on top of the navigation system (Fig. 2) plus several offboard processes for managing the website (Fig. 5). The **task sequencing** layer is responsible for carrying out Xavier's tasks. This includes commanding the path planning layer to navigate to the requested goal location, centering the robot at the doorway if the destination is an office or classroom, and executing the given task (taking a picture, saying "hello," telling a knock-knock joke). Currently, the task sequencing layer


File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Netsite:
5316
5315
5317
5312
5311
5309
5310
5307
5304
5403
5305
5301
5302
5404
http://jubilee.learning.cs.cmu.edu:8080/locationZoomi

Status as of Sep 15, 15:43 EDT: Xavier is not responding.

Xavier is usually on line 2 to 4 pm EST (19:00 to 21:00 GMT) on weekdays with odd-numbered dates (i.e., alternating weeks of M-W-F and T-Th). This schedule may vary somewhat to allow Xavier to assist in other research.

Tell Xavier Where To Go!

View Options (options ordered from high to low fidelity/bandwidth):

- Update using Server Push with higher-resolution images (~34K) every 5 seconds
- Update using Server Push with lower-resolution images (~16K) every 10 seconds
- Current: Uses Client Pull with higher-resolution images (~32K) every 20 seconds when Xavier is operating, every 3 minutes when it is off-line)
- Update using Client Pull with lower-resolution images (~14K) every 20 seconds when Xavier is operating, every 2 minutes when it is off-line)

Figure 4. *Xavier monitoring web page.*

has only limited ability to monitor for task achievement or to recover from failures. Related work, however, has developed a much more sophisticated task sequencing layer [4, 11].

The **communications bridge** process, which resides onboard, is responsible for exchanging data between the onboard and offboard processes over a radio modem. Data exchange is via the TCA message-passing facility, which is built on top of TCP-IP. The bridge process receives task and data requests from the offboard processes and forwards them to the appropriate onboard processes. In the other direction, the communications bridge receives position estimates, route plans, and camera images (gifs) from the onboard processes and forwards them to the offboard processes.

By having one process responsible for all onboard/offboard communications, Xavier autonomously (and safely) continues to carry out its tasks, even if it loses communication with the outside world.

The rationale for having one process responsible for all onboard/offboard communications is that if radio communication is lost for a while, the other onboard processes are not blocked trying to send data. In this way, Xavier autonomously (and safely) continues to carry out its tasks, even if it loses communication with the outside world (which occurs more often than we care to think about).

The web-site management system consists of two offboard processes, running on a Sparc5 workstation, that interface to a Netscape web server, also running on the Sparc machine. The **task manager** is responsible for queuing user requests, dispatching requests to the task sequencing layer (via the communications bridge), and sending e-mail confirmations to users after each task is completed. The task manager uses a simple scheduling algorithm that tries to minimize the time until users' requests are executed. It computes the utility of going to a particular location as the sum of the utilities for each pending request for that destination, where the utility of an individual request is an exponential function of how long the request has been pending. The task manager then chooses the destination with the highest utility. Thus, it will be indifferent between a destination for which a single user has been waiting a fairly long period of time and one where many users have been waiting shorter periods. Note that, in particular, there has been no effort to minimize the overall travel distance of the robot, since the original goal of the experiment was to stress-test the navigation system.

The **web manager** process is responsible for maintaining the web pages. It requests position information and camera images, creates a gif showing the robot in the map, and creates new web pages with the robot status, map, and camera images. It also creates new command interface pages, depending on which floor Xavier is currently on (or which floor it will be on when it next runs). The web manager actually creates four types of pages that differ in the bandwidth requirements needed for viewing on the web. The pages with the lowest bandwidth requirements contain a low-resolution camera image and are updated every 20 seconds. The pages with the highest bandwidth requirements have a high-resolution image and are updated continually with streaming images ("push" technology).

While the robot is on-line infrequently, due to battery limits and other research demands for its use, the task manager and web manager processes are always running. If the web manager cannot connect (via TCA and the communications bridge) to the onboard processes, it assumes the robot is off-line and adjusts the status message accordingly. When the task manager receives a request and Xavier is off-line, it queues the request. In this way, users can get access to the robot, eventually, even if they are unable to connect to it during normal operational hours (due to time zone differences, etc.)

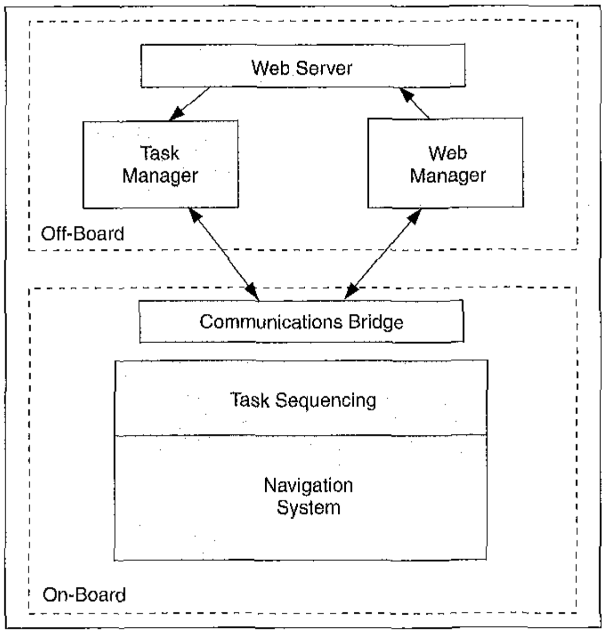


Figure 5. *Offboard and onboard system.*

Lessons Learned

The main lesson learned was about the reliability of the navigation system. During a three-year period of web-based operation (December 1995 through December 1998), Xavier received over 30,000 requests and carried out over 4700 separate tasks (since requests are queued and then bundled together, the number of tasks is smaller than the total number of requests). In the process, Xavier operated for over 340 hours and traveled over 210 kilometers (Fig. 6).

The average success rate for achieving that set of tasks was about 95%, and that has increased to about 98% in recent months (Fig. 7; see also [11] for a discussion of the navigation results). We also learned that nothing beats having naive users to test a system's reliability. One example: the first day we put Xavier on the web, the software crashed repeatedly. The reason was that people were requesting Xavier to go to where it already was, and we had never tested that capability before. While the fix was simple, it nonetheless gave us renewed respect for the need to test thoroughly, and in an unbiased manner.

From the perspective of web-based robotics, we learned lessons that stemmed from the facts that the robot was both mobile and autonomous. The robot's mobility had both positive and negative impacts on web interactions. The positive effect (gleaned through comments in our guestbook) was that users felt that controlling a mobile robot remotely was a unique experience.

However, many of the effects of mobility on connecting with the web were negative, especially as compared to stationary web-based robots, such as those described elsewhere in this issue. The need for radio communication limits the bandwidth to the robot, which lessens the interactivity that can be achieved. Running on batteries limits the on-line time of the robot to a few hours per day. Even though the web interface is always operational, the fact is that most web visitors do not see Xavier in action when they happen to visit its site. This is exacerbated by the fact that, in the past year, we connect Xavier to the web less frequently—at this point, it is on-line less than once a week.

Probably the most severe effect of mobility on web-based interaction is a sociological one: Users can see what Xavier sees, but they cannot see Xavier itself. This is often disorienting, especially since the images are updated only every few seconds (higher bandwidth would definitely help). The Rhino tour-guide robot [1] overcame this problem by using an overhead camera to track the robot, which was feasible in their case because they operated in an environment where the robot was usually in view of a single camera. Another approach would be to use a pair of robots, each watching the other. One of the visitors to the website suggested having a full-length mirror on one of the walls and making that one of Xavier's destinations, so that people can command Xavier to go and look at itself.

On the other hand, the fact that Xavier is autonomous had mostly positive effects on web-based interactions. For one, autonomy mitigated the effects of low bandwidth and unreliable communication. Since the robot is being tasked at a high level (traveling to discrete locations), high-bandwidth interaction is not strictly necessary. Even if communication is lost completely, Xavier can still continue achieving its current task. In particular, none of the navigation components is affected by loss of communication, so the robot's safety (and that of the people it encounters) is not affected. When communication is restored, the offboard processes reconnect with the onboard communications bridge, often automatically, and usually without need to restart processes.

The only real negative impact of autonomy on web-based interaction is that commanding at a high level is not as interactive as teleoperation. Some users have expressed an interest in being able to choose an arbitrary location on the map for Xavier to go. Although the navigation system can handle that, for logistical reasons we do not want to allow that level of control. In particular, many occupants of our building are not too keen on having the robot visit them and tell them jokes on a regular basis.

The only real negative impact of autonomy on web-based interaction is that commanding at a high level is not as interactive as teleoperation.

One of the more surprising lessons learned was the degree to which people accept Xavier at face value. Given the nature of the web, it would be comparatively simple to "fake" Xavier's travels with a series of canned images and a simple simulator (much simpler, probably, than creating an autonomous mobile robot). For the most part, however, few web visitors have ever questioned the authenticity of the robot. One exception occurred early on. Since Xavier uses a probabilistic navigation scheme, with a spatial resolution of one meter, it sometimes stops near, but not actually at, its destination. In such cases, the pictures e-mailed back to requesters would show walls rather than doors or open offices. Occasionally, we would get back responses questioning whether Xavier was really doing what it claimed. We solved this by training a neural net to recognize visually when the camera was pointed towards a doorway and to use a simple visual servoing routine to move the robot directly in front of the door. Since implementing this extension, we have not received any more comments about whether the robot is real.

An especially popular aspect are the knock-knock jokes Xavier tells (popular with web visitors, not so much with the

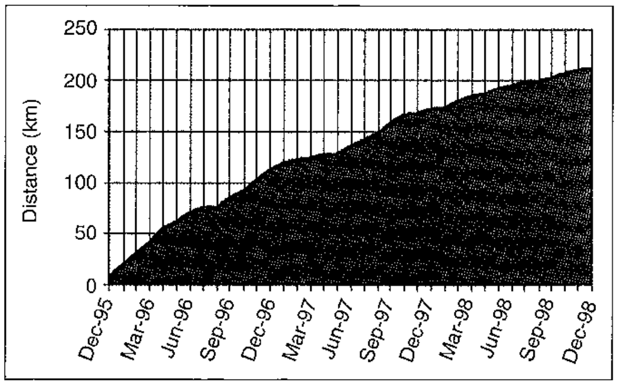


Figure 6. *Cumulative travel distance for Xavier.*

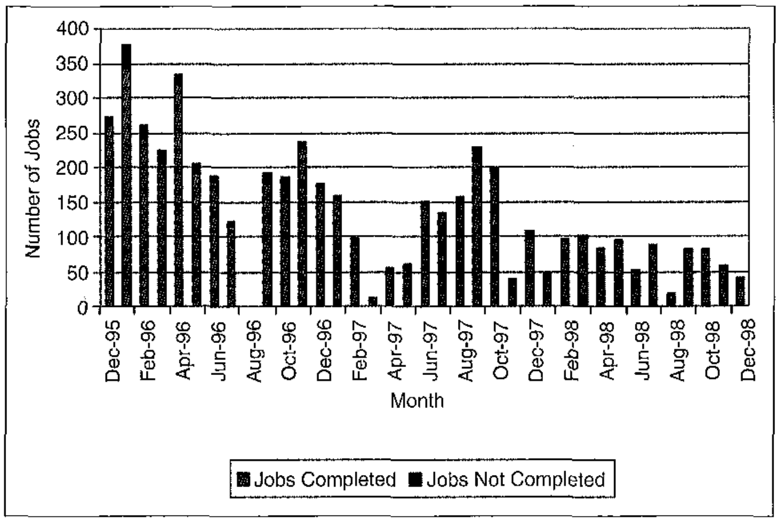


Figure 7. *Tasks completed and successfully performed.*

occupants of our building). We created a "jokes contest" web page for people to submit knock-knock jokes that involve Xavier (example: "Knock knock" - "Who's there?"; "Xavier" - "Xavier who?"; "Zave-yer self from these awful jokes, turn me off"). New jokes continue to be submitted, even after four years, testifying to the collective creativity on the web. Some visitors have even suggested allowing users to submit arbitrary messages for Xavier to say at its destination. Imagine the sociological consequences of that on the residents of our building! Sometimes creativity can be taken a bit too far.

Based on our experience, we have a number of observations that can guide the implementation of future web-based robots. The most important is the need for high-quality feedback. When we first constructed the web interface to Xavier in 1995, one priority was to minimize the bandwidth used, so that the web interface would not interfere with other projects. The result is a rather slow refresh rate (5-10 seconds), which makes it difficult to see what Xavier is doing.

Since the original design, Xavier's computational power has tripled and standardized low-bandwidth mechanisms and protocols such as Java and RealVideo have been developed and become ubiquitous. It is now possible, with a low computational overhead to Xavier, to generate a continuous low-bandwidth, real-time video feed. Similarly, it is possible to construct dedicated Java applets so that map and position information can be displayed rapidly and efficiently (for instance, Minerva uses such a mechanism effectively [13]).

An important aspect of the web-based experience is user interaction. Robotic websites can facilitate this by providing ways for users to relate their experiences. For instance, the Telegarden project [3] hosts interactive "chat rooms" for discussions related to robotics, technology, and the web. In Xavier's case, we provide a guestbook where users can leave comments. Reading Xavier's guestbook, we see many indications that an autonomous robot on the web strikes a particular chord with audiences not often associated with robots:

"I am 4 and my name is Alexander. I am going to be 5 in 2 weeks and I want a robot for my birthday to clean up my room and play pinch attack. ... I would like to play games with you on the computer. I have to go to bed now, we are leaving now. Thank-you and goodbye." - Alex T., March 7, 1998.

"This is fantastic! I'm new to the web and feel like a kid in a toy store for the first time. I happen to be 54 years old." - Mary H., October 9, 1998.

Unfortunately, not every visitor is pleased with the experience. By far, the most frequent complaint is from visitors who miss those few hours when Xavier is live on the web (especially users in very different time zones). We have tried to alleviate this in several ways, including sending e-mail to notify users when the tasks are completed. We are also considering notifying users a few minutes before their queued requests are to be undertaken, to give them a chance to see Xavier live. However, none of this solves the fundamental problem that the web demands immediate feedback—continuous 24-hour presence is an important goal for future web based robots.

Conclusions

Overall, our web-based robot experiment has been very successful. It has conclusively demonstrated the reliability of our navigation system, has given our robot project very good publicity, and has introduced many people around the world to the wonders (and limitations) of autonomous mobile robots. While the scientific results of the experiment have long since been achieved, we have no intention of putting a halt to this experiment in interactive, web-based robotics.

Acknowledgments

Xavier is the result of the collaborative efforts of many people. Special mention goes to Lonnie Chrisman, Domingo Gallardo, Karen Zita Haigh, Nak Yong Ko, and Sebastian Thrun. Greg Armstrong has worked tirelessly to maintain Xavier and to make sure it carries out its appointed rounds. Thanks also to the thousands of web visitors who have watched and commanded Xavier over the years.

Keywords

Web-based robotics, autonomous mobile robots, reliable indoor navigation

References

[1] W. Burgard, A. Cremers, D. Fox, D. Hahnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "The interactive museum tour-guide robot," in *Proc. National Conf. Artificial Intelligence*, Madison, WI, 1998, pp. 11-18.

[2] J.L. Fernandez and R. Simmons, "Robust execution monitoring for navigation plans," in *Proc. Intelligent Robots and Systems (IROS)*, Victoria, BC, Canada, 1998, pp. 551-557.

[3] K. Goldberg, J. Santarromana, G. Bekey, S. Gentner, R. Morris, C. Sutter, and J. Wiegley, "A telerobotic garden on the World Wide Web," *SPIE Robotics and Machine Perception Newsletter*, vol. 5, no. 1, Mar. 1996.

[4] K.Z. Haigh and M.M. Veloso, "Interleaving planning and robot execution for asynchronous user requests," *Autonomous Robots*, vol. 5, no. 1, pp. 79-95, Mar. 1998.

[5] S. Koenig, R. Goodwin, and R. Simmons, "Robot navigation with Markov models: A framework for path planning and learning with limited computational resources," in *Reasoning with Uncertainty in Robotics* (vol. 1093 of *Lecture Notes in Artificial Intelligence*), Dorst, van Lambalgen, and Voorbraak, Eds. New York: Springer-Verlag, 1996, pp. 322-327.

[6] S. Koenig and R.G. Simmons, "Xavier: A robot navigation architecture based on partially observable Markov decision process models," in *Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems*, D. Kortenkamp, R. Bonasso, R. Murphy, Eds. MIT Press, 1998, pp. 91-122, 1998.

[7] S. Koenig and R.G. Simmons, "Unsupervised learning of probabilistic models for robot navigation," in *Proc. Int. Conf. Robotics and Automation*, Minneapolis MN, Apr. 1996, pp. 2301-2308.

[8] N.Y. Ko and R. Simmons, "The Lane-Curvature Method for local obstacle avoidance," in *Proc. Intelligent Robots and Systems (IROS)*, Victoria, BC, Canada, 1998, pp. 1615-1621.

[9] R. Simmons, "Structured control for autonomous robots," *IEEE Trans. Robotics and Automation*, vol. 10, Feb. 1994.

[10] R. Simmons, "The Curvature-Velocity Method for local obstacle avoidance," in *Proc. Int. Conf. Robotics and Automation (ICRA)*, Minneapolis, MN, 1996, pp. 3375-3382.

[11] R. Simmons, R. Goodwin, K.Z. Haigh, S. Koenig, and J. O'Sullivan, "A Layered architecture for office delivery robots," in *Proc. Autonomous Agents 1997*, Marina del Rey, CA, Feb. 1997, pp. 245-252.

[12] R. Simmons and S. Koenig, "Probabilistic robot navigation in partially observable environments," in *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, Montreal, Quebec, Canada, 1995, pp. 1080-1087.

[13] S. Thrun, M. Bennewitz, W. Burgard, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz, "MINERVA: A second generation mobile tour-guide robot," in *Proc. Int. Conf. Robotics and Automation (ICRA)*, Detroit, MI, March 1999.

Reid Simmons is a senior research scientist in the School of Computer Science at Carnegie Mellon University. He earned his B.A. degree in 1979 from SUNY at Buffalo and his M.S. and Ph.D. degrees from MIT in 1983 and 1988, respectively, in the field of artificial intelligence. His thesis work focused on the combination of associational and causal reasoning for planning and interpretation tasks. Since coming to Carnegie Mellon in 1988, Dr. Simmons' research has focused on developing self-reliant robots that can autonomously operate over extended periods of time in unknown, unstructured environments. This work involves issues of robot control architectures that combine deliberative and reactive control, decision-theoretic planning, robust execution monitoring and error recovery, and indoor and outdoor navigation. His current interests include multirobot coordination and human-robot social interaction.

Joaquin L. Fernandez was born in Lugo, Spain, in 1968. He received the M.S. degree in telecommunications engineering from the University of Vigo, Spain, in 1992. He spent one year as a visitor researcher in the Department of Computer Science at Carnegie Mellon University. Currently, he is an assistant professor in the Departamento de Ingenieria de Sistemas y Automatica at the University of Vigo, where he recently completed his Ph.D. degree. His current research interests include supervision, fault diagnosis, and navigation in mobile robots.

Richard Goodwin is a researcher at the IBM T.J. Watson Research Center in Yorktown Heights, NY. He was awarded a B.Sc. in electrical engineering from the University of Waterloo in 1987 and earned M.S. and Ph.D. degrees in artificial intelligence from the School of Computer Science at Carnegie Mellon University in 1994 and 1996, respectively. His Ph.D. thesis on "Meta-Level Control for Decision-Theoretic Planners" developed techniques for effective planning with limited computational resources and applied these techniques to controlling mobile robots and medical diagnosis planning. Since joining IBM, his work has focused on agent-based decision-support for optimizing supply-chains and business to business e-commerce.

Sven Koenig is an assistant professor in the College of Computing at the Georgia Institute of Technology. He received his Ph.D. degree from Carnegie Mellon University for his dissertation on "Goal-Directed Acting with Incomplete Information." His research centers around techniques for decision making (planning and learning) that enable situated agents to act intelligently in their environments and exhibit goal-directed behavior in real-time, even if they have only incomplete knowledge of their environment, limited or noisy perception, imperfect abilities to manipulate it, or insufficient reasoning speed. His research draws on areas such as artificial intelligence, robotics, decision theory, and operations research. He has written over 30 publications in these areas and is the recipient of various awards and fellowships, including the Tong Leong Lim Pre-Doctoral Prize of the University of California at Berkeley and the Raytheon Faculty Fellowship of Georgia Institute of Technology.

Joseph O'Sullivan received a B.A. (Mod) in computer science from Trinity College Dublin, Ireland, in 1992 and an M.Sc. in computer science from Carnegie Mellon University in 1997. He is currently a Ph.D. candidate in computer science at Carnegie Mellon University, where he is a member of the Learning Robot Laboratory. His main interest is in how robots can learn from small amounts of data by making use of domain-specific knowledge. This research is manifesting itself in a thesis on the curriculum aspects of training robots and other agents over time. More generally, he is interested in machine learning, autonomous agents, robotics, and perception.

Address for Correspondence: Reid Simmons, Carnegie Mellon University, School of Computer Science, Pittsburgh, PA 15214. Tel: +1 412 268 2621. Fax: +1 412 268 5576. E-mail: reids+@cs.cmu.edu.