Xavier: An Autonomous Mobile Robot on the Web

Reid Simmons, Joaquin Fernandez[1], Richard Goodwin[2],
Sven Koenig[3], Joseph O'Sullivan

School of Computer Science, Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

For the past three years, we have been running an experiment in web-based interaction with an autonomous indoor mobile robot. The robot, 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 over 30,000 requests received and 210 kilometers travelled, 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.

Introduction

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. Now, three years, 30,000 requests and 210 kilometers 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 web site (http://www.cs.cmu.edu/~Xavier).

The mobile robot, Xavier (Figure 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 on-board computers are connected to each other via thin-wire Ethernet.

Figure 1: Xavier

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 are about 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 tour guide robot [1] is another, highly successful, web-based autonomous mobile robot). 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

1. Now at University of Vigo, Vigo, Spain.
2. Now at IBM T J Watson Research Center, Yorktown Heights, NY.
3. Now at Georgia Institute of Technology, Atlanta, GA.

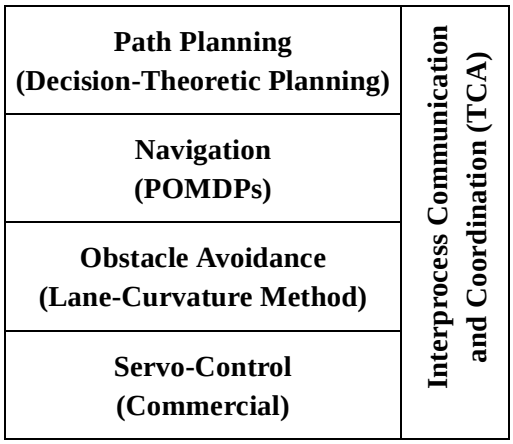

Figure 2: The Navigation System

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 introduces problems of its own, particularly in the area of interactivity. People seem to prefer "hands on" control, and do not get the same type of immediate feedback with an autonomous mobile robot that they would with a teleoperated one. 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 web site.

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).

The next section describes the on-board navigation software system of Xavier. The following section describes the off-board interface to the web, and how it interfaces to Xavier. We then present some lessons learned over the past three years with regards to autonomous mobile robots and web-based interaction.

Autonomous Navigation System

The Xavier navigation system is a layered architecture (Figure 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 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.

The **path planning** layer determines efficient routes based on a topological map, 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, that 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 non-roboticists to interact

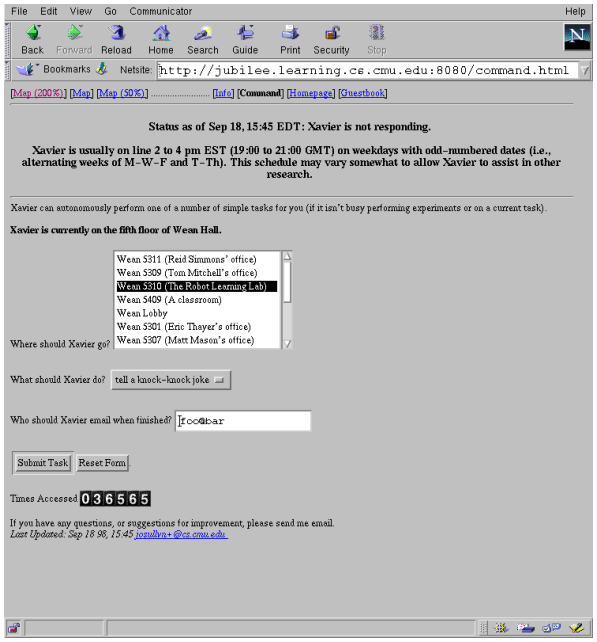

Figure 3: Command Interface Web Page

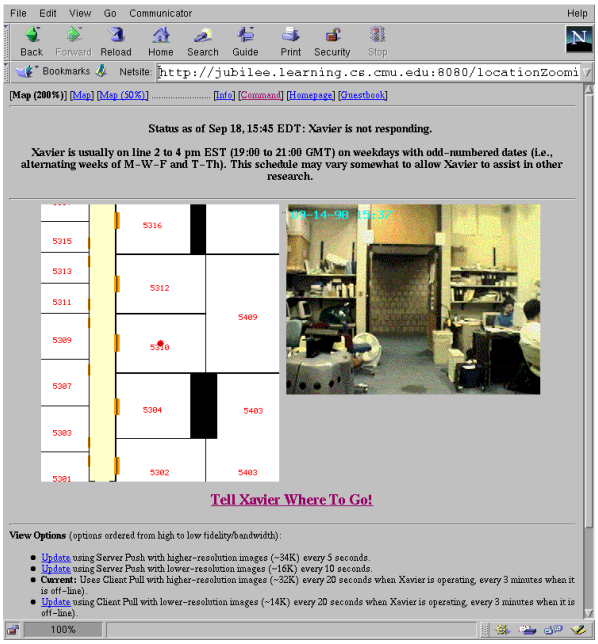

Figure 4: Xavier Monitoring Web Page

with the robot. The command interface web page (Figure 3) shows Xavier's current status (updated every 5-10 seconds) and 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) email address. 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 the user submits the task request, a confirmation web page is sent back immediately 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 email address, Xavier will send email after it achieves the task, and 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 (Figure 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 an additional on-board layer on top of the navigation system (Figure 2) plus off-board processes for managing the web site (Figure 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 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 also resides on board, is responsible for exchanging data between the on-board and off-board 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 off-board processes and forwards them to the appropriate on-board processes. In the other direction, the communications bridge receives position estimates, route plans and camera images (gifs) from the on-board processes and forwards them to the off-board processes.

The rationale for having one process responsible for all on-board/off-board communications is that if radio communication is lost for a while, the other on-board 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.

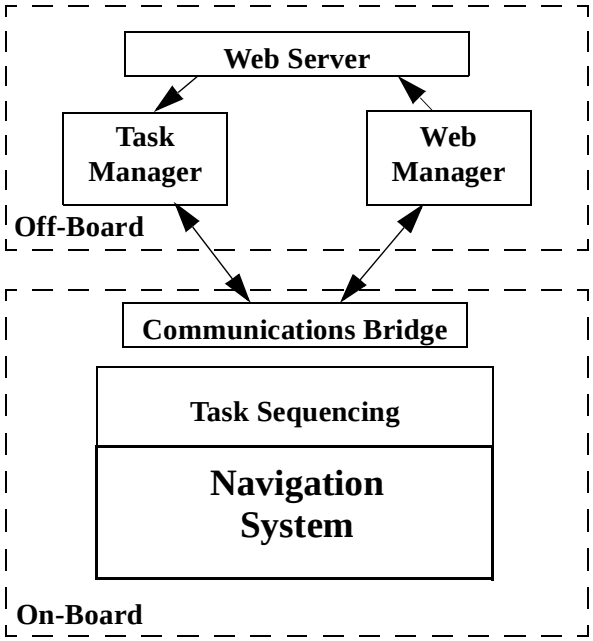

Figure 5: Off-Board and On-Board System

The web site management system consists of two off-board 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 queueing user requests, dispatching requests to the task sequencing layer (via the communications bridge), and sending email 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 get 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 the 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 only in the bandwidth requirements needed for viewing on the web. The pages with the lowest bandwidth

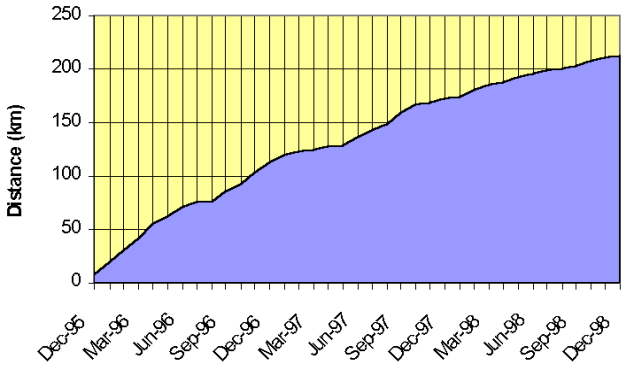

Figure 6: Cumulative Travel Distance for Xavier

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 use of the robot, the task manager and web manager processes are always running. If the web manager cannot connect (via TCA and the communications bridge) to the on-board 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 timezone differences, etc.)

Lessons Learned

The main lesson learned was about the reliability of the navigation system. During three years of web-based operation (December 1995 through December 1998), Xavier received over 30,000 requests and carried out over 4,700 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 (Figure 6). The average success rate for the past three years in achieving tasks is about 95%, and that has increased to about 98% in recent months (Figure 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

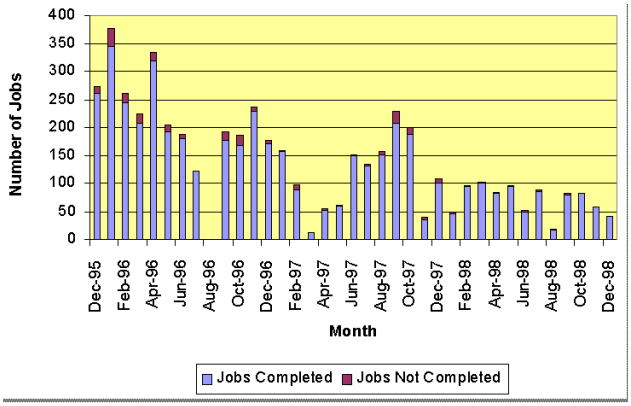

Figure 7: Tasks Attempted & Successfully Performed

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. 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 web site (this is exacerbated by the fact that, in the past year, we have taken Xavier on line less frequently -- it is now down to about once a week). The need for radio communication limits the bandwidth to the robot, which lessens the interactivity that can be achieved.

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 very 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 the single camera. Another approach would be to use a pair of robots, each watching the other. One of the visitors to the web site 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 are 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 off-board processes reconnect with the on-board communications bridge, often automatically and usually without need to restart processes. Although, from the perspective of user interaction, this is an advantage only if communication is lost for short periods of time, as is in fact the case with Xavier's wireless network.

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.

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 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 emailed 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 then using a simple visual servoing routine to move the robot directly in front of the door. After this extension was implemented, we did not receive any more comments about whether the robot was real.

An especially popular aspect are the knock-knock jokes Xavier tells (popular with web visitors, not so much with the 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 three 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 second), 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 part of the feedback mechanism is a guestbook where users can leave comments. This is invaluable, both for gauging the diversity and range of users and for soliciting suggestions. In addition to the usual collection of scatological and self-promotional comments, there are indications in Xavier's guestbook 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. My cat's name is Zoe. I liked seeing pictures of you. I go to Brookview Montessori and I do the hundred board. 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.

Taking this a step further, robotic web sites should host interactive "chat rooms" for discussions related to robotics, technology and the web. Such mechanisms have been used with great success by the Tele-Garden project [3].

Some users want more technical information about the robot, including details on robot construction and programming. To accommodate this, we constructed a complete web site around our original web interface. We find, however, that there are always unanswered questions. While it is time-consuming, it is important to keep the informational web pages accurate and current for disseminating technical information to the public at large.

By far, the largest complaint is from users 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 email 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.

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.

References

[1] W. Burgard, A. Cremers, D. Fox, D. Hahnel, G. Lakemeyer, D. Schulz, W. Steiner, S. Thrun. The Interactive Museum Tour-Guide Robot. in *Proc. National Conference on Artificial Intelligence*, pp 11-18, Madison WI, 1998.

[2] J. L. Fernandez and R. Simmons. Robust Execution Monitoring for Navigation Plans. In *Proc. Intelligent Robots and Systems (IROS)*, Victoria Canada, 1998.

[3] K. Goldberg, J. Santarromana, G. Bekey, S. Gentner, R. Morris, C. Sutter, J. Wiegley. A Telerobotic Garden on the World Wide Web. *SPIE Robotics and Machine Perception Newsletter*, **5:1**, March 1996.

[4] K. Z. Haigh, M. M. Veloso. Interleaving Planning and Robot Execution for Asynchronous User Requests. *Autonomous Robots*, **5:1**, pp 79-95, March 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 Dorst, van Lambalgen and Voorbraak, eds. *Reasoning with Uncertainty in Robotics*, volume 1093 of *Lecture Notes in Artificial Intelligence*, pp 322-327, Springer. 1996.

[6] S. Koenig, 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, pp 91-122, 1998.

[7] S. Koenig, R.G. Simmons. Unsupervised Learning of Probabilistic Models for Robot Navigation, In *Proceedings of the International Conference on Robotics and Automation*, pp 2301-2308, Minneapolis MN, April 1996.

[8] N.Y. Ko and R. Simmons. The Lane-Curvature Method for Local Obstacle Avoidance. In *Proc. Intelligent Robots and Systems (IROS)*, Victoria Canada, 1998.

[9] R. Simmons. Structured Control for Autonomous Robots. *IEEE Transactions on Robotics and Automation*, **10:1**, Feb. 1994.

[10] R. Simmons. The Curvature-Velocity Method for Local Obstacle Avoidance. In *Proc. of the International Conference on Robotics and Automation (ICRA)*, pp 3375-3382, 1996.

[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 '97*, pp 245-252, Marina del Rey, CA, February 1997.

[12] R. Simmons and S. Koenig. Probabilistic Robot Navigation in Partially Observable Environments. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp 1080-1087, Montreal Canada, 1995.

[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. of the International Conference on Robotics and Automation*, March 1999.