Published in Proc. of the 1999 IEEE/RSJ International Conference on Robots and Systems (IROS'99), pp. 942-947, Kyongju, Korea 1999

Robust Visualization of Navigation Experiments with Mobile Robots over the Internet

Dirk Schulz[†] Wolfram Burgard[‡] Armin B. Cremers[†]

[†]Department of Computer Science III, University of Bonn, 53117 Bonn, Germany
[‡]Department of Computer Science, University of Freiburg, 79110 Freiburg, Germany

Abstract

Visualization is an important precondition for successful tele-operation of instructable mobile robots. Data connections with varying and limited bandwidth such as the Internet, however, prohibit the continuous transmission of video signals. In this paper we propose a predictive simulation technique which is designed to permit the reliable visualization of the robot's actions over the Internet. It differs from previous approaches in that it includes an odometry and sensor simulation. This simulation of the robot allows the integration of a complete robot control system to reliably predict complex actions of the robot even if large transmission gaps of several seconds occur. We describe an application of the predictive simulation technique to navigation experiments with mobile robots. We present different experiments carried out with a real robot illustrating that the predictive simulation technique provides accurate visualizations of the robot's actions even if transmission gaps of more than ten seconds occur.

1. Introduction

Due to the increased costs of laboratory equipment such as mobile robots and due to the increased specialization of research on the other hand, there is a growing need for cooperation between research groups. Visualization over the Internet will become an important means for the collaboration of research groups, for example during the development and the testing of control systems or for the demonstration of new systems.

Unfortunately the Internet does only provide a restricted bandwidth and arbitrary large transmission gaps can occur so that video streams cannot be used for the visualization of the robot's actions. Obviously, accurate visualizations are important for the observation of experiments and during demonstrations. Furthermore, with the development of web-operated robots, for example tele-operated museum tour-guide robots, there is a growing need for tele-operation interfaces which can be used even by untrained people. In this context, accurate visualizations of the robot's actions become more and more important.

In this paper we describe a predictive simulation scheme (PSS) which provides a basis for smooth and reliable real-time 3D visualizations of the movements of an autonomous mobile robot based on an object-oriented 3D CAD environment model. Predictive simulations are employed to synchronize the visualization with the current state of the robot even during larger transmission gaps of several seconds.

The long term behavior of an autonomous robot is usually determined by the planning component of the robot's control system. Thus, to make long term predictions, it is necessary to include this planner into the predictive simulation. The PSS therefore provides a robot simulator and appropriate interfaces to integrate a control system. It furthermore is able to exploit a 3D model of the environment for the visualization of the robot's actions as well as for the simulation of proximity sensors.

We built a prototypical implementation of PSS for our mobile robot Rhino. In the current system, the path planning component and the collision avoidance of Rhino's control system are integrated into the PSS. The experiments we carried out with our system show that the PSS-based visualization is highly accurate even in situations in which transmission gaps of over ten seconds occur. We believe that such techniques are a basic precondition for a robust visualization of navigation experiments over the Internet. This will allow distributed research groups to carry out and observe joint experiments with expensive laboratory equipments such as mobile robots.

The remainder of this paper is organized as follows. After discussing related work in the next section, Section 3 introduces the PSS framework. Section 4 briefly describes the parts of the Rhino system, which have been integrated into our system. Finally, Section 5 presents experimental results which illustrate the strength of the approach.

2. Related Work

A variety of web-based tele-operation interfaces for autonomous robots has been developed over the last few years. Three of the earlier systems are the Mercury Project, the "Telerobot on the Web" and the Tele-Garden [7, 8, 16]. These systems allow people to perform simple tasks with a robot arm via the web. They provide still images from a camera mounted on the robot arm after a requested movement task has been completed, no feedback is given while the robot arm is moving.

XAVIER [15] is a web-operated autonomous mobile robot. It can be advised by web users to move to an office and to tell a joke. The web interface relies on client-pull and server-push techniques to provide images taken by the robot and a map indicating the robot's current position.

Hirukawa et al. [9] describe web-operation interfaces, where web users can perform manipulation tasks using a 3D graphics simulation contained in the web browser. These interfaces follow the tele-programming approach. Tasks are first tested in a simulation and afterwards executed by the real robot.

Simulation based delay compensation techniques and virtual environment interfaces are in use for space and undersea tele-robotics for several years now [4, 19, 14, 1]. These methods are designed to deal with large transmission delays but, in contrast to Internet based systems, rely on communication links with a guaranteed bandwidth and known transmission gaps. A visualization method, which is often used in tele-manipulation systems are predictive displays [11]. Methods of this type combine low-bandwidth video transmissions with graphics visualizations.

The simulation scheme we use is conceptually similar to the one used during the ROTEX experiment on board the space-shuttle COLUMBIA in 1993 [10]. During this experiment a 6 DOF robot arm was controlled from ground employing tele-sensor programming. A human operator planned a sequence of "elemental operations" which was afterwards carried out by the robot. Elemental operations are short-term reactive control tasks, like for example grasping operations, which the robot can carry out autonomously using its sensors. Truly autonomous robots, however, include methods to plan even complex actions and to react to unforeseen circumstances. For example, in environments which are not completely accessible by the robot, which is generally the case, the robot must have the ability to determine the shortest path from its current trajectory to a given target point and to re-plan dynamically whenever unexpected obstacles block the robot's path.

The predictive simulation system described in this paper is designed to provide a robust visualization of navigation experiments carried out with mobile robots over the Internet. The main advantage of the PSS is that it allows accurate and smooth visualizations of the robot's actions even over unreliable connections without a guaranteed bandwidth such as the Internet. In the case of transmission gaps, the PSS uses a robot simulator to simulate the robot's actions. It furthermore provides interfaces to integrate a robot control system into the PSS. This way, the PSS can reliably visualize complex navigation experiments even if transmission gaps of more than ten seconds occur.

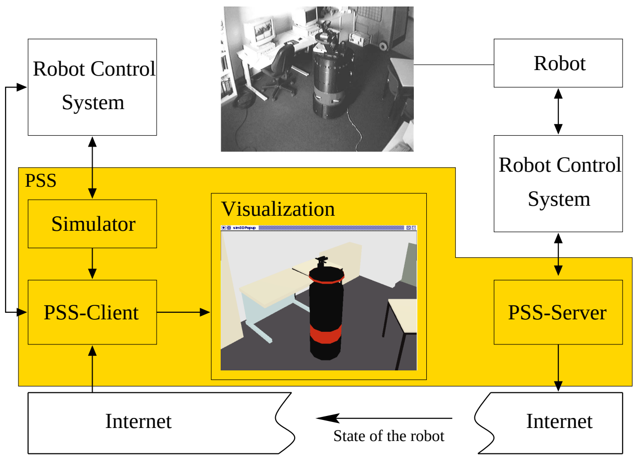


Figure 1. Architecture of the PSS.

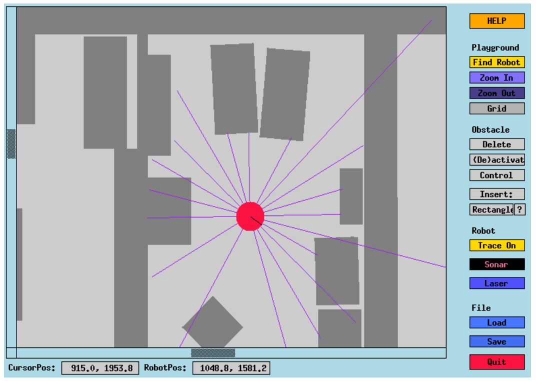
3. The Predictive Simulation Scheme PSS

The task of PSS is to achieve a better visualization accuracy over the Internet when long transmission gaps caused by high packet loss rates occur. To compensate lost packets and to provide smooth visualizations during transmission gaps, the PSS provides simulators to simulate the robot's sensors, a 3D-visualization component of the robot in its environment, a network synchronization mechanism, which receives the state information from the real robot via Internet, and interfaces allowing to establish a communication between the robot control system and the PSS. The architecture of the PSS and the major flow of information is shown in Figure 1.

3.1. Sensor Simulation

To achieve fast visualizations of the robot's actions and to perform a simulation of the robot's sensors, PSS uses a 3D environment model. The model is organized as a hierarchical scene graph, where objects are described in the boundary representation. Based on this representation, the PSS renders this scene graph several times per second.

The PSS furthermore provides methods for carrying out ray casting within the world model which builds the basis for the proximity sensor simulation. Realistic scene descriptions generally contain a large number of objects so that a huge amount of line-polygon intersections have to be calculated to simulate the proximity sensors. To achieve this



CursorPos: 915.0, 1953.8 RobotPos: 1048.8, 1581.2

Figure 2. The simulator provides a 2D visualization of the robot and the computed sonar measurements.

simulation in real-time, the PSS applies a spatial indexing technique based on the rectangular tiling of the scene [13].

To establish a realistic simulation of proximity sensors the PSS takes sensor specific measuring errors into account. For example, ultrasound is subject to specular reflection. In case of a reflection the sensor does not perceive an echo and thus erroneously indicates free space. According to experiments carried out with the ultrasound sensors of our mobile robot Rhino, the probability of such a reflection increases with the angle α between the surface normal and the acoustic axis of the beam and also depends on the distance d to the obstacle. We therefore approximate the probability $P(r \mid \alpha, d)$ of a specular reflection by

$$P(r \mid \alpha, d) = \begin{cases} 0 & \alpha < \theta(d) \\ (\alpha - \theta(d))/\beta & \theta(d) \leq \alpha < \theta(d) + \beta \\ 1 & \alpha \geq \theta(d) + \beta \end{cases}$$

where β denotes the angle range in which the failure probability increases linearly from 0 to certainty, and $\theta(d)$ is a decreasing linear function. The values of β and $\theta(d)$ depend on the material the obstacle is made of. An example of a typical scan generated in an office environment is shown in Figure 2. For the more reliable laser-range finder we use the distance to the next obstacle in the sensing direction and add an appropriate Gaussian noise to model the small measuring errors.

3.2. Network Synchronization

In order to synchronize the visualization with the real robot, the PSS-server sends synchronization messages via the Internet to the PSS-client. Since the Internet is subject to packet loss, the use of a reliable protocol like TCP causes arbitrary long transmission delays, because the retransmission of lost messages prevents the arrival of new information. In several experiments carried out with typical long distance connections, we frequently observed packet loss rates of up to 30 percent. In contrast to that, the transmission time of a single packet rarely exceeded 0.3 seconds, and the accuracy of the re-synchronization of the PSS-client solely depends on the transmission time of the latest synchronization message. To minimize the time required to obtain the latest packet, the PSS uses the unreliable UDP protocol. Each synchronization message therefore includes a state vector completely describing the state of the robot at time t as well as the time-stamp t when this vector was sent. The PSS-client extrapolates the state of the robot at message arrival time and warns the user if a packet needed more than 0.4 seconds for the transmission. The system relies on the "Network Time Protocol" [12] to synchronize the clocks of the computers involved for this purpose.

After receiving a synchronization message the PSS-client transmits it to the robot control system, which itself uses the PSS-simulation to compute the next actions of the robot. In return, the PSS-client receives the state of the robot in the simulation. Since this predictive simulation runs within a local network, state messages can be delivered at high frequency which is the basis for the smooth visualizations.

4. A Predictive Simulation for the Rhino System

Using the PSS, we implemented a prototype of a visualization tool for our mobile robot Rhino [2, 18]. Rhino is an RWI B21 synchro-drive robot equipped with 2 laser range finders, a ring of 24 ultrasonic sensors as well as with several tactile and IR sensors. In this prototype, the PSS uses the reactive collision avoidance component and the path planner of Rhino's control system for the predictive simulation. All components including the network interface of the PSS communicate via local socket connections using TCX [5] as the communication server. In the following we briefly introduce the components of the Rhino control system.

Path Planning: Rhino's path planner uses value iteration [18] to determine the shortest path from the robot's position to its goal location. Path planning takes place in a grid map representation of the environment, which is computed from the 3D model through projection. Value iteration determines the optimal path towards the target point for every position in the environment. After that, depending on the robot's position, the optimal path to the goal position is extracted. The path planner of the Rhino system computes a sequence of intermediate "target points" which are transmitted to the collision avoidance system

Collision Avoidance: Rhino uses the dynamic window approach [6] for reactive collision avoidance. The collision

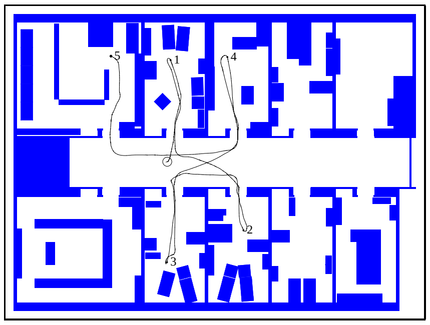


Figure. 3. Trajectory of the robot during the office delivery task.

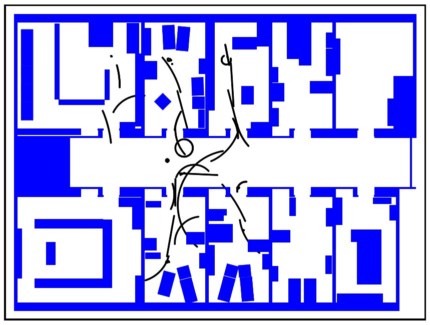


Figure 4. Trajectory estimated by extrapolation.

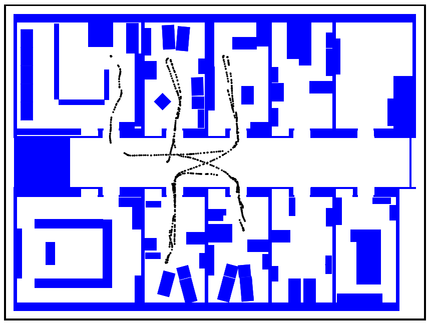


Figure 5. Trajectory predicted by PSS.

avoidance controls the velocities of the robot based on sensor input. Its task is to move the robot to a “target location” prescribed by the path planner. To avoid collisions with obstacles it takes the dynamics of the robot into account.

Given these components it suffices to transfer the robot's current position, its current velocities and accelerations, as well as the current target point of the path planner to the PSS-client in order to realize the visualization.

5. Experimental Results

To demonstrate the prediction accuracy of the PSS, we performed several navigation experiments with our mobile robot Rhino. The goal of these experiments is to demonstrate the advantages of the PSS over a visualization technique which extrapolates the most recent state information received from the PSS-server.

In the first experiment, Rhino started in the corridor of our department and visited five different offices in the order which is depicted in Figure 3. The length of the trajectory is 80.8 m and the robot needed 320 seconds to complete this task. The numbers in the offices correspond to the target-points sent to the path planning module.

To evaluate the capabilities of the PSS, we manually decreased the packet transmission rate to approximately 10 packets per minute. We used a constant transmission interval, so that the time gap between subsequent packets was approximately 6 seconds. Since packets sometimes get lost, the effective time-gap between subsequent packets in several cases was $n \times 6$ seconds. Figure 4 shows the trajectory obtained if the robot's movements are extrapolated based on the most recent packet. Obviously the resulting trajectory differs largely from the true trajectory of the robot. It furthermore contains several long gaps of more than $1m$ illustrating the limited quality of the standard extrapolation approach. The PSS, however, provides a trajectory which is much closer to the robot's actual trajectory (see Figure 5). Thus, the PSS provides a better estimation of the robot's position than the extrapolation approach.

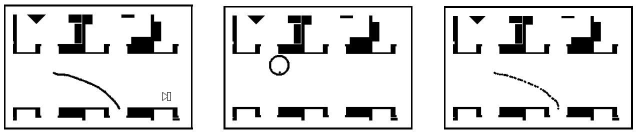
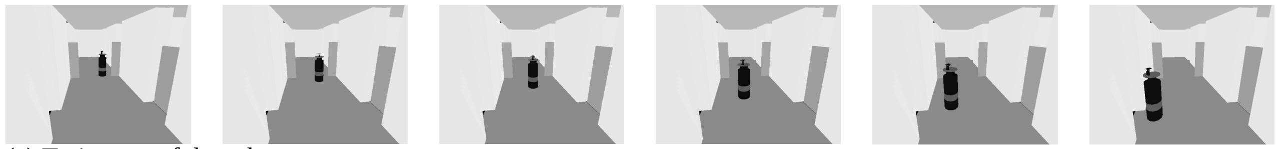


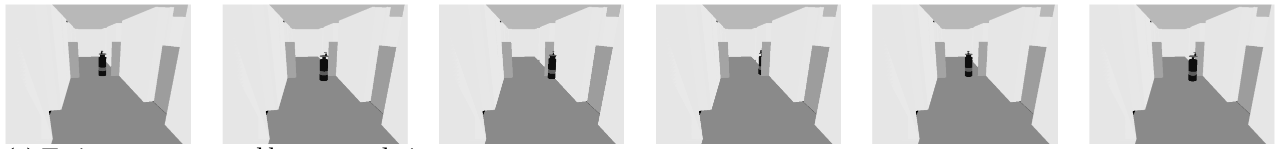
Figure 6. Trajectory of the robot taken after leaving room 1 (left), circular trajectory obtained by extrapolation (center), and trajectory predicted by PSS (right).

To illustrate the improvements in the visualization we now consider a small fraction of the trajectory of the robot after it left room 1. Figure 6 shows the corresponding trajectory of the robot (left image), the trajectory obtained by extrapolation (center image), and the trajectory computed by the PSS (right image). In this particular case one packet was lost so that the system had to predict the robot's trajectory for a period of 12.5 seconds. Since the extrapolation does not change the velocities of the robot during a transmission gap, the robot moves on a circular trajectory. In contrast, the PSS uses the path planner and the collision avoidance modules to compute intermediate velocities. Therefore, the resulting trajectory is much closer to that of the real robot.

The corresponding sequences of computed images for the 3D visualization are shown in Figure 7. The time delay between consecutive frames is 2 seconds. The first row shows the images obtained using the correct position of the robot. The second row contains the images computed by the extrapolation, and the third row includes the images obtained using the PSS. Obviously, the PSS significantly improves the quality of the visualization.

We repeated this experiment several times given the data recorded during this run. In each experiment we randomly selected the packets which were not transmitted. Figure 8 shows the average positioning error – the distance between the predicted position and the position of the real robot – as a function of transmission gap. This shows, that the PSS is able to significantly reduce the average positioning error compared to the extrapolation technique after transmission gaps of at least 2.5 seconds. Since these differences depend


(a) Trajectory of the robot.


(c) Trajectory computed by extrapolation.

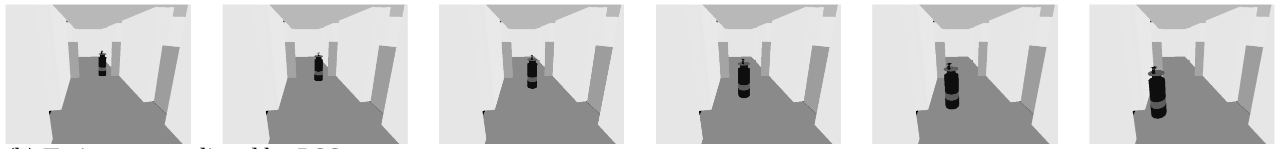

(b) Trajectory predicted by PSS.

Figure 7. A sequence of computed images showing Rhino moving through the corridor of our department. The time-difference between consecutive images is approximately 2 seconds. The camera position is illustrated in the left image of Figure 6.

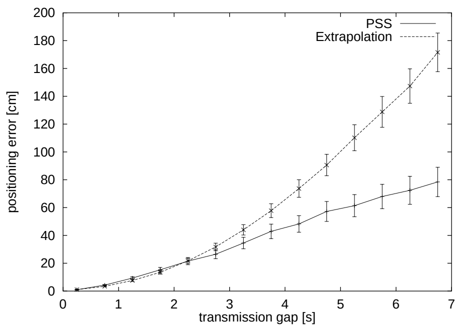


Figure 8. Average displacement depending on the length of the transmission gap.

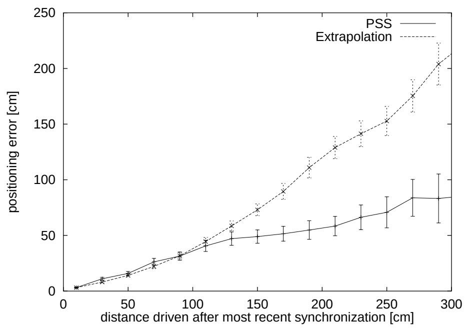


Figure 9. Average displacement depending on the distance traveled after the last synchronization.

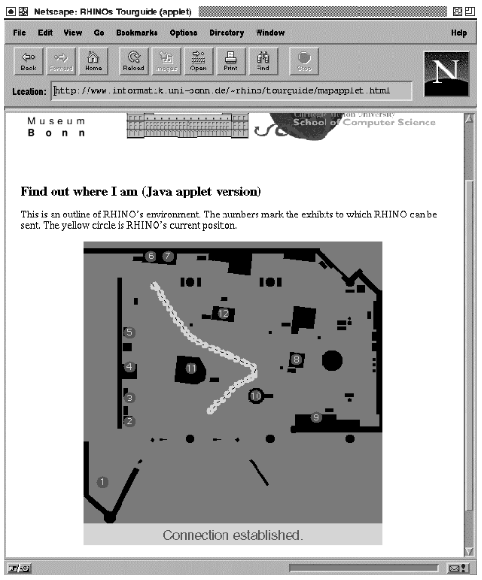
on the speed of the robot, which was $34.5\,cm/s$ on average in this experiment, we additionally computed the average displacement depending on the distance traveled after the latest synchronization packet. The resulting plots are shown in Figure 9. Thus, at least after $1m$ of travel, PSS provides a significantly better prediction than extrapolation. In both figures, the error bars indicate the 95% confidence interval of the average mean.

Recently, we used parts of the PSS during the deployment of two mobile robots as interactive museum tour-guides which also possessed Web interfaces allowing to remotely control them over the Internet [3, 17]. Both Web-Interfaces (Figure 10 illustrates Rhino's interface) included a Java-applet for a smooth animation of the robots movements in a 2D-projection of the environment. During the 19 days of the deployments of the two robots more than 4000 virtual visitors used this applet to monitor the activities of the robots.

6. Summary and Conclusions

In this paper we presented the PSS, a predictive simulation system for the accurate visualization of navigation actions of autonomous mobile robots over the Internet. The PSS has a client-server architecture. The server is linked to the robot control system and broadcasts the status of the robot to all its clients. The clients use the PSS-simulator and a copy of the robot control system to predict the actions of the robot after receiving a status message. This way, the PSS can reliably visualize the robot's action even if transmission gaps of over 10 seconds occur.

The PSS has been implemented and tested in a typical office environment using the mobile robot Rhino. Extensive



http://www.informatik.uni-bonn.de/~rhino/tourguide/mapapplet.html

Find out where I am (Java applet version)

This is an outline of RHINO's environment. The numbers mark the exhibits to which RHINO can be sent. The yellow circle is RHINO's current positon.

Connection established.

Figure 10. Java-applet based on the PSS displaying smooth trajectories of the robot Rhino

experiments showed that PSS is robust against transmission gaps of up to several seconds and that it provides smooth and realistic visualizations even in such situations. Parts of the PSS have furthermore been used during the deployment of mobile robots Rhino and Minerva as interactive museum tour-guides to visualize the movements of the robot over the Internet.

Despite these encouraging results there are several warrants for future research. The most important topic concerns the extension to dynamic environments and the visualization of manipulation tasks. This requires a reliable model update mechanism based on the sensor interpretation capabilities provided by the robot system.

References

[1] B. Bon and S. Homayoun. Real-time model-based obstacle detection for the NASA ranger telerobot. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, April 1997.

[2] J. Buhmann, W. Burgard, A. Cremers, D. Fox, T. Hofmann, F. Schneider, J. Strikos, and S. Thrun. The mobile robot Rhino. *AI Magazine*, 16(2):31–38, Summer 1995.

[3] W. Burgard, A. B. Cremers, D. Fox, G. Lakemeyer, D. Hähnel, D. Schulz, W. Steiner, and S. Thrun. The interactive museum tour-guide robot. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 1998.

[4] L. Conway, R. A. Volz, and M. W. Walker. Teleautonomous systems: Projecting and coordinating intelligent action at a distance. In *IEEE Transactions on Robotics and Automation*, volume 6, April 1990.

[5] C. Fedor. TCX. An interprocess communication system for building robotic architectures. Programmers guide to version 10.xx. Carnegie Mellon University, Pittsburgh, PA, December 1993.

[6] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, Mar. 1997.

[7] K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, and J. Wiegley. Desktop tele-operation via the world wide web. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1995.

[8] K. Goldberg, J. Santarromana, G. Bekey, S. Gentner, R. Morris, J. Wiegley, and E. Berger. The telegarden. In *Proc. of ACM SIGGRAPH*, 1995.

[9] H. Hirukawa, T. Matsui, and H. Onda. Prototypes of teleoperation systems via a standard protocol with a standard human interface. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1997.

[10] G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl. Sensor-based space robotics–ROTEX and its telerobotic features. In *IEEE Transactions on Robotics and Automation,* volume 9, October 1993.

[11] W. S. Kim and A. K. Bejczy. Demonstration of a high-fidelity predictive/preview display technique for telerobotic servicing in space. In *IEEE Transactions on Robotics and Automation*, volume 9, October 1993.

[12] D. L. Mills. Network time protocol (version 3) specification, implementation and analysis. In *Request for Comments (RFC) 1305, Internet Engineering Task Force (IETF),* March 1992.

[13] H. Samet. *Applications of Spatial Data Structures*. Addison-Wesley Publishing Company, 1990.

[14] T. B. Sheridan. Space teleoperation through time delay: Review and prognosis. In *IEEE Transactions on Robotics and Automation*, volume 9, October 1993.

[15] R. Simmons, R. Goodwin, K. Haigh, S. Koenig, and J. O'Sullivan. A layered architecture for office delivery robots. In *Proceedings of the First International Conference on Autonomous Agents*, Marina del Rey, CA, February 1997.

[16] K. Taylor and J. Trevelyan. A telerobot on the world wide web. In *Proceedings of the 1995 National Conference of the Australian Robot Association*, July 1995.

[17] S. Thrun, M. Bennewitz, W. Burgard, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. MINERVA: A second-generation museum tour-guide robot. unpublished.

[18] S. Thrun, A. Bücken, W. Burgard, D. Fox, T. Fröhlinghaus, D. Hennig, T. Hofmann, M. Krell, and T. Schmidt. Map learning and high-speed navigation in RHINO. In D. Kortenkamp, R. Bonasso, and R. Murphy, editors, *AI-based Mobile Robots: Case studies of successful robot systems*. MIT Press, Cambridge, MA, 1997.

[19] Y. Tsumaki and M. Uchiyama. A model-based space teleoperation system with robustness against modeling errors. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, April 1997.