Controlling Mobile Robots with Distributed Neuro-Biological Systems

Alfredo Weitzenfeld
Comp Eng Dept, ITAM
Rio Hondo 1, San Angel Tizapan
Mexico City, MEXICO, 01000
Email: alfredo@itam.mx

Sebastian Gutierrez-Nolasco
School of Inf and Comp Science
University of California, Irvine
Irvine, CA 92697-3430, USA
Email: seguti@ics.uci.edu

Nalini Venkatasubramanian
School of Inf and Comp Science
University of California, Irvine
Irvine, CA 92697-3430, USA
Email: nalini@ics.uci.edu

***Abstract*— Nature has always been a source of inspiration in the development of autonomous robotic systems. As such, the study of animal behavior (ethology) and the study of the underlying neural structure responsible for behavior (neuroethology) have inspired many robotic designs. In general, the complexity of these behaviors has a direct impact on robot efficiency. For example, behaviors involving neural network based adaptation and learning can become very inefficient under real-time processing constraints. To overcome these constraints, autonomous mobile robots need either powerful hardware, or alternatively, have to be linked to distributed grid-like computer networks using wireless communication. While the first approach simplifies the overall robotic architecture, it results in larger and more expensive robots. On the other hand, the second approach results in smaller and inexpensive robots, although involving more complex distributed architectures requiring wireless communication capabilities. The work presented in this paper discusses the challenges in modeling autonomous robots inspired by biological systems and our approach to embedding mobile robots to distributed computational resources.**

I. Introduction

Many different approaches have been proposed to the control of autonomous robots including architectures inspired in biology. There are two general approaches to biologically inspired robots, one based on ”ethology” (animal behavior) and the other one based on ”neuroethology” (behavior related to neurobiological structure) [1]. While both approaches have inspired different robotic architectures, the neuroethological approach has the additional benefit of enabling the experimentation of otherwise simulated-only neuroscientific modeling. By providing with a robotic experimentation platform, many issues that are oversimplified in simulation can be further analyzed during embodiment. However, an important concern with neuroethological robotic experimentation is the requirement of real-time performance, a critical issue considering the expensive nature of neural processing. While powerful ”super” robots could be built, our particular approach has been to embed smaller and inexpensive robots with low power requirements into remote networks of computers via wireless communication. Under such a computing architecture time-consuming processing can be done remotely in general purpose computers with the advantage of friendly programming environments that need not be reimplemented into the robots. A number of such Internet embedded systems have been developed [2] including many teleoperated robotics [3].

In the case of autonomous robots, time-consuming tasks can take advantage of the distributed embedded architecture by doing ”off-board” computing, as in the case of neural networks and image processing. However, there are a number of issues that arise from the need to achieve real-time performance with distributed architectures, such as slow communication, unreliable transmission, limited bandwidth, disconnectivity and completely failures. To take advantage of distributed embedded architectures it is first necessary to overcome restrictions in wireless communication. For such purpose, we have developed a distributed embedded robot architecture supported by adaptive middleware managing communication, as well as local and remote resources.

The work presented in this paper discusses the challenges in modeling autonomous robots inspired by biological systems and our approach to embedding mobile robots to distributed computational resources. We describe our current work in conducting neuroethological robotic experimentation using the MIRO (Mobile Internet Robotics) system linked to the NSL (Neural Simulation Language) [4] system with communication managed by an Adaptive Robotic Middleware (ARM).

II. Embedded Distributed Architecture

Developing software for autonomous mobile robots is an error-prone task due to a large number of different methods for capturing and processing sensor information (multi-resolution and multi-granularity) and the lack of efficient management of communication (quality of service, bandwidth, security) in the presence of routinely interference and failures. As an initial effort towards our goal, we have developed the MIRO system as shown in Figure 1. The architecture consists of multiple robots, each one connected via wireless communication to its own particular copy or instance of the neural computational system. Processing is distributed among the robotic hardware and the remote computational system. While it is possible to share robot ”intelligence” among multiple robots, in our particular applications we have kept the robots ”fully” autonomous in reproducing neuroethological experimentation. Other applications could easily take advantage of information sharing (see [5]). Under our architecture: (i) time-consuming processes are carried out in the (neural) computational system, implemented using the distributed NSL system, (ii) sensory input, motor output and other limited tasks are carried out in the robot hardware and (iii) communication and data transformation is managed by the ARM framework.

A typical computation cycle involves the robot initially sending sensory input (visual and tactile) data to the neural computational system through the ARM framework, which optimizes the flow of information (back and forth). The neural computational system then processes the sensory input cycling through its neural modules while finally sending motor output back to the robot. These cycles continue indefinitely or until some specific task is accomplished. In such a way, the computational system provides the robot's "intelligence", while the robot does limited "on-board" processing.

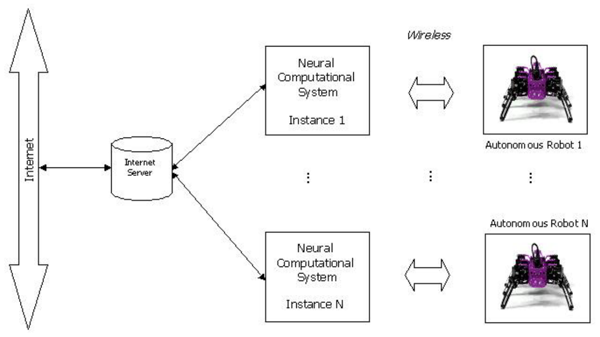

Internet
Internet Server
Wireless
Neural Computational System Instance 1
Neural Computational System Instance N
Autonomous Robot 1
Autonomous Robot N

Fig. 1. MIRO embedded robotic architecture consisting of multiple autonomous robots linked to their own instance of the distributed neural computational system. All such instances are connected to Internet and might be monitored remotely

A. Adaptive Robotic Middleware

The great advantage of a middleware approach is that it provides with transparent mechanisms to enhance application response at run-time by automatically reconfigure itself in order to respond to changes in the environment, a critical aspect in our embedded real-time architecture, where resource and power constraints pervade the application. In such a way, the ARM framework allows specification of communication requirements in a high level manner that can be later associated with low-level specific architectural implementations using a comprehensive set of basic communication protocols. That is, the ARM is responsible in determining *how*, *when* and *what* information should be modified in order to match communication fluctuations. For example, robots use captured video as sensory input to visuomotor coordination behavior, thus bandwidth adaptation is needed to deliver the information in a manner sensitive to resources available, and entails the use of media conversion and compression to achieve the desired results.

The ARM architecture consists of two key components: (i) the *communication manager*, which provides and enforces application level communication requirements, and (ii) the *adaptation manager*, which provides adaptation mechanism operating at different levels of abstraction. In fact, adaptation can be reactive or proactive. Reactive adaptation is triggered when failure to achieve intended communication goals is detected; while proactive adaptation results from detecting that a better or more efficient communication can be achieved under the current environment conditions. Hence, when a robot is deployed, it is equipped with performance monitors that detect whether or not expected communication requirements are met. If a failure to meet these requirements is consistently repeated, the communication manager determines its probable cause and, depending on its cause, a new set of requirements are generated using higher level adaptation strategies (supplied by the adaptation manager) such as increasing or decreasing the level of the service.

Since specifics of a communication protocol may pose conditions on the system environment that prevent or constraint its combination with other communication protocols, we leverage from our previous work [6] a generic rule base, which is used as an oracle to determine which protocol implementation is suitable for a particular situation in terms of coverage and efficiency. Within this oracle, each communication protocol specification is enhanced with (i) a list of prerequisites or dependencies of the protocol in terms of other protocols, (ii) a list of restrictions that prevent the protocol to be composed with other protocols, and (iii) a list of interaction parameters that can be tuned to assure proper interaction with other protocols.

The communication manager is structured in five components: (1) the oracle, (2) a possible set of communication protocols used to customize the communication between the robot and the computational system, (3) a protocol installer/uninstaller module capable to install or uninstall communication protocols dynamically, (4) a protocol loader module capable to initialize or save the state of the communication protocol in case of installation or un-installation (respectively) and (5) a resident communication module running in the robot.

Every protocol installed as well as the resident communication module have performance monitors and a set of parameters that can be tuned to improve their performance. In the robot, every communication is logged with specific information that is (eventually) shipped to the adaptation manager, which consolidate the information with the measurements taken in the computational system and cross-reference it with the battery monitors (robot and camera) measurements in order to estimate power consumption and communication performance based on communication patterns. Strategies and policies are encoded in a pattern repository used by the adaptation manager as a helper to determine possible actions in case a change is required (whether is reactive or proactive).

Figure 2. shows the relationship between the communication and the adaptation manager components using a secure compressed video capture as a communication requirement. In the figure, the communication manager receives the communication requirement and consults the protocol oracle to select the appropriate communication protocols and their correct interaction parameter values. When the interaction parameters have been set, it request the protocol installer to install the corresponding protocols and the protocol loader to instantiate the protocols with the specified interaction parameters. Once the protocol is running, performance monitors gather information that the adaptation profiler uses to determine if a modification is required. In case an adaptation is needed, the adaptation profiler consult the adaptation repository, fetchs the most suitable strategy or pattern to be applied and sends it to the communication manager to be implemented.

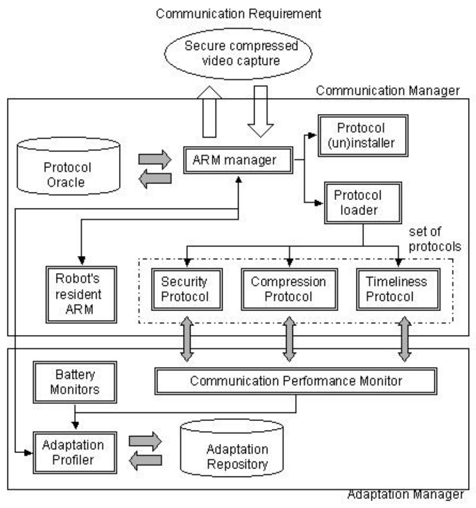


Fig. 2. Communication and adaptation manager components and their interaction using a secure compressed video capture requirement.

III. Biologically inspired Mobile Robots

As part of our effort to get an understanding of the underlying biological mechanisms involved in living organisms we have been followed a multi-level approach to brain theoretic neural modeling [7]. We have developed many neurobiological models for frogs and toads, praying mantis and monkeys, where models distinguish among high-level behavior and low-level neural structure [8].

At the behavioral level, neuroethological data from living animals is gathered to generate single and multi-animal systems to study the relationship between a living organism and its environment, giving emphasis to aspects such as cooperation and competition between them. Examples of behavioral models include the praying mantis Chantlitaxia (”search for a proper habitat”) [9] and the frog and toad prey acquisition and predator avoidance models [10]. We describe behavior in terms of perceptual and motor schemas [11] representing a hierarchical and distributed model for action-perception control.

At the structural level, neuroanatomical and neurophysiological data are used to generate perceptual and motor neural network models corresponding to schemas developed at the behavioral level. These models try to explain the underlying mechanisms for sensorimotor integration. Examples of neural network models are the toad's tectum-pretectum-thalamus responsible for discrimination among preys and predators [12], the prey acquisition and predator avoidance neural models [13] and the toad prey acquisition with detour behavior model involving adaptation and learning [14]. Neural networks are processed via the Neural Simulation Language NSL [4]. Models that involve neural networks are usually limited in scope as in [15], while more complex models are simplified in terms of their inherent neural complexity [16]. For example, let us consider the toads ”prey-predator” visuomotor coordination model described in [8], with schema and neural level components shown in Figure 3.

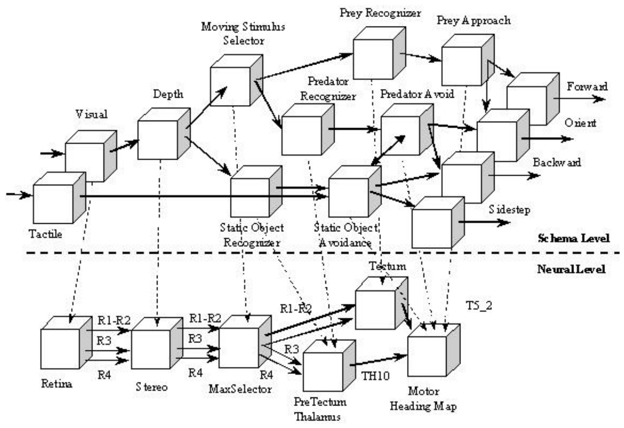


Fig. 3. Toad's prey-predator visuomotor coordination model architecture with schema and neural level modules. At the schema level, blocks correspond to schemas or behavior agents representing animal or robot behavior. At the neural level, blocks represent neural networks, some having a direct correspondence to brain regions

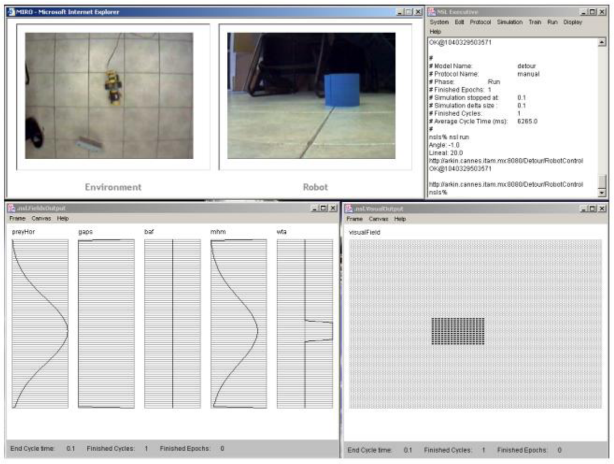
IV. Experiments and Results

As part of the process of robot experimentation we have modeled prey acquisition and predator avoidance neuroethological behaviors initially simulated under NSL where their correctness is first tested. After that, the models are prototyped under the MIRO robot architecture to test their behavior under real world conditions. To monitor system results, Internet-linked aerial cameras as well as robot cameras were included, as show in Figure 4 (top), with neural behaviors visualized (down) as the actual experiments are performing. Obviously there is an additional penalty to pay in performance when doing real time visualization, although it is well worth during model development or fine-tunning. The MIRO architecture has proven quite beneficial in providing real-time monitoring capabilities of robot behavior.

Figure 5 shows a sample output for one of the experiments, involving prey acquisition with a 10cm barrier showing direct detour. The experiment was carried out on a single Lego-based robot connected in a wireless fashion to the MIRO system. A wireless camera was added on top of the robot transmitting video in a wireless fashion to remote video capture server using the ARM framework. The video capture server is located in the computational system, but not necessarily in the same domain where NSL is running. In this scenario, the ARM is divided in a resident ARM module (running in the robot) and a video receiver ARM module co-allocated with the video capture server.

Since the video data is used as a sensory input to determine the robot's behavior, timeliness of delivery of the video data is required to assure that any action taken based on this information will not compromise the robot's (safety) operation. Thus, the resident ARM module minimizes video transfer latency by modifying the video capture parameters such as frames per second, image quality, and compresses the video data before sending it to the ARM video receiver, which needs to convert the data before delivering it to the image processing component for object identification and localization. Object identification is achieved by dividing the image into (color based) regions, which we call region of interest. Since each region may contain multiple objects, filters are applied to the image within each region in order to detect color discontinuities. Once the objects have been identified and classified, the information is transformed to a suitable format for further processing by NSL.

Initially, we set up the prey-like stimulus to ”blue” and Predator-like stimulus to ”red”. Thus, every blue object recognized by the robot will generate an attraction field represented as a positive stimulus; while every red recognized object will generate a repulsive field represented as a negative stimulus. For the time been, static object recognition will generate a passive field represented as zero stimulus that only will change to a positive value if the robot's collision sensor is activated. Table I shows the average power consumption of the video capture using three different configurations used in the experiments.



```
OK@1040329503671

#
# Model Name:                  detour
# Protocol Name:               manual
# Phase:              Run
# Finished Epochs:  1
# Simulation stopped at:       0.1
# Simulation delta size :      0.1
# Finished Cycles:             1
# Average Cycle Time (ms):     6265.0
#
nsls% nsl run
Angle: -1.0
Lineal: 20.0
http://arkin.cannes.itam.mx:8080/Detour/RobotControl
OK@1040329503671

http://arkin.cannes.itam.mx:8080/Detour/RobotControl
nsls%
```

```
End Cycle time:   0.1   Finished Cycles:   1   Finished Epochs:   0
```

```
End Cycle time:   0.1   Finished Cycles:   1   Finished Epochs:   0
```

Fig. 4. Top: Internet aerial view of autonomous robot and robot's camera view of ”blue” prey-like stimulus. Down: NSL frames showing results from different visual and neural modules in a basic prey acquisition robot experiment (without barrier).

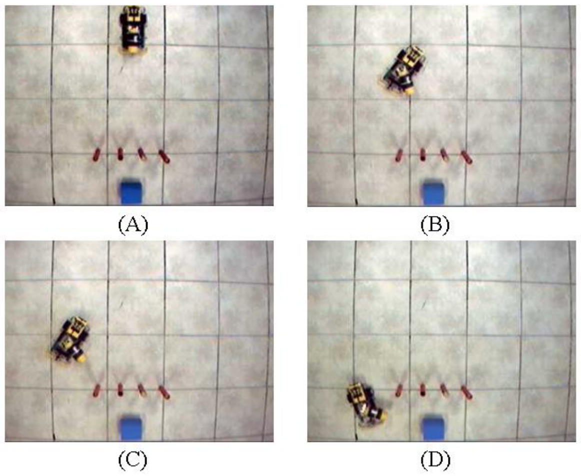


Fig. 5. Results from prey acquisition experiment for 10cm barrier with direct detour around barrier.

TABLE I
AVERAGE POWER CONSUMPTION OF THE VIDEO CAPTURE USING DIFFERENT CONFIGURATIONS.

Frame Resolution	*Frames per Second*	*Avg. Power (W)*
320x240	15	5.86
340x160	15	5.60
160x120	15	5.30

Among the interesting aspects that have emerged from the robot experiments is the problem of ”losing” the prey once the robot directs itself around the barrier, something that was not considered in the original simulated models. While this was solved by a ”pan” control on the camera, where the camera can always ”look” into the prey, it raises an interesting number of issues such a recalling prey positions from memory such as with memory saccade models [17].

V. Related Work and Concluding Remarks

In the past years a number of research efforts have been carried out to link autonomous robots with Internet, such as Xavier [18], Rhino [19], Minerva [20], KephOnTheWeb [21], as well as others [22]. These efforts have highlighted the potential of Internet when applied to autonomous robots (and robotics in general) as well as some of its limitations. The benefits are direct access (control and monitoring) to sometimes expensive and/or remotely located equipment, such as with the Mars Pathfinder Sojourner Rover [23]. While many such robots are teleoperated, the possibility to link autonomous robots to the Internet has the benefit of being able not only to monitor its behavior but also enhance it by linking it with other specialized systems having additional processing resources.

The work presented here overviews the inherent complexity in modeling autonomous robots inspired by biological systems both in terms of behavior and structure. This complexity can be managed by having a multi-level architecture emphasizing both top-town and bottom-up designs where robot behaviors can be implemented by neural network processing when available. In general, one of the main concerns with neural networks has been the expensive nature of computation, in particular, when involving adaptive behavior [14]. To improve on performance and reduce the size and cost of robots, we have developed an embedded distributed robot architecture. While most time-consuming tasks can take advantage of the distributed robotic system by processing them remotely, there are a number of ”behavioral” or ”task” issues that arise from such a distributed architecture. The main consideration is what happens when communication between the robot and computational system fails or becomes extremely slow. This may be caused by a number of potential problems, such as lack of wireless service, low bandwidth, switching among multiple access points, low power or even failures in robot hardware. While some situations may cause a complete failure of the robot, others could be handled. In fact, the robot could respond in many ways to a failure in communication ranging from simply waiting without doing anything until communication is restored, ending its mission, up to performing limited tasks that may put it back in action, such as the ”Chantlitaxia” model [9].

The experimental results were both remarkable and encouraging, considering the simplicity of the methods used for feature detection and extraction (color-based regions of interests). Further experiments are needed in order to estimate the robustness of object identification under varying lightning conditions and invariance regarding object backgrounds.

Until now, the general approach has been to process each neural level module in a different machine, while schema level modules are assigned to the machine close to the corresponding neural modules they communicate with. Yet, new challenges arise in integrating neural modules, such as the use of different temporal scales, something that requires additional considerations.

ACKNOWLEDGMENT

The authors would like to thank the UC MEXUS CONACYT Advanced Network Services Collaborative Grant: ”MIRO: Adaptive Middleware for a Mobile Internet-based Robot Laboratory” as well as the ”Asociacin Mexicana de Cultura, A.C.” and the different students at the CANNES Laboratory at ITAM who participate in this project.

REFERENCES

[1] Arkin, R.C, *Behavioral based Robotics.* MIT Press, 1998.

[2] Goldberg, K. and Siegwert, R., *Beyond Webcams: An Introduction to Online Robots.* MIT Press, 2002.

[3] Sukhatme, G.S. and Mataric, M.J., “Embedding Robots Into the Internet,” *Communication of the ACM*, vol. 43(5) pp 67-73, 2000.

[4] Weitzenfeld, A., Arbib, M. and Alexander, A., *The Neural Simulation Language: A System for Brain Modeling.* MIT Press, 2002.

[5] Balch, T. and Arkin, R.C., “Communication in Reactive Multiagent Robotic Systems,” in *Autonomous Robots*, 1994.

[6] Gutierrez-Nolasco, S. and Venkatasubramanian, N., “A Reflective Middleware Framework for Communication in Dynamic Environments,” in *International Symposium on Distributed Objects and Applications*, 2002.

[7] Arbib, M.A, “Levels of Modeling of Mechanisms of Visually Guided Behavior,” in *Behavior Brain Science 10:407-465*, 1987.

[8] Weitzenfeld A., “A Multi-level Approach to Biologically Inspired Robotic Systems,” in *10th International Conference on Artificial Neural Networks and Intelligent Systems*, 2000.

[9] Arkin, R.C., Ali, K., Weitzenfeld, A. and Cervantes-Perez, F., “Behavior Models of the Praying Mantis as a Basis for Robotic Behavior,” *Robotics and Autonomous Systems*, vol. 32 (1) pp. 39-60, 2000.

[10] Cobas, A. and Arbib, M.A., “Prey-catching and Predator-avoidance in Frog and Toad: Defining the Schemas,” *Theoretical Biology*, vol. 157, 271-304, 1992.

[11] Arbib, M.A., *Schema Theory, in the Encyclopedia of Artificial Intelligence, Editor Stuart Shapiro, 2:1427-1443*, 2nd ed. Wiley, 1992.

[12] Cervantes-Perez, F, Lara, R. and Arbib, M.A., “A neural model of interactions subserving prey-predator discrimination and size preference in anuran amphibia,” *Theoretical Biology*, vol. 113, 117-152, 1985.

[13] Cervantes-Perez, F., Herrera, A. and Garca, M., “Modulatory effects on prey-recognition in amphibia: a theoretical ’experimental study’,” *Neuroscience: from neural networks to artificial intelligence, Editors P. Rudoman, M.A. Arbib, F. Cervantes-Perez, and R. Romo*, vol. (4):426-449, 1993.

[14] Corbacho, F. and Arbib M., “Learning to Detour,” in *Adaptive Behavior, 3(4):419-468*, 1995.

[15] Fagg, A., King, I, Lewis, A., Liaw, J. and Weitzenfeld, A., “A Testbed for Sensorimotor Integration,” in *IJCNN, 1:86-91*, 1992.

[16] Weitzenfeld, A., Cervantes, F. and Sigala, R., “NSL/ASL: Simulation of Neural based Visuomotor Systems,” in *International Joint Conference on Neural Networks*, 2001.

[17] Dominey, P. and Arbib, M.A., “A cortico-subcortical model for generation of spatially accurate sequential saccades,” *Cerebral Cortex*, vol. 1(2):153-175, 1992.

[18] Simmons, R., Fernndez, J., Goodwin, R., Koening, S. and Sullivan, J. , “Lessons Learned from Xavier, Ongoing Experiments in Interactive Web based Robotics with an Autonomous Mobile Robot on the Web,” *IEEE Robtics and Automation Magazine*, vol. (7)1: 33-39, 2000.

[19] Bugard, W., Cremers, A., Fox, D., Habnel, D., Lakemeyer, G., Schulz, D., Steiner, W. and Thrun, S., “The interactive museum tour-guide robot,” in *15th National Conference on Artificial Intelligence*, 1998.

[20] Thrun, S., Bennewitz, M., Bungard, W., Cremers, A., Dellaert, P., Fox, D., Habner, D., Rosenberg, C., Schulte, J. and Shulz, D., “MINERVA: A second-generation museum tour-guide robot,” in *International Conference on Robotics and Automation*, 1999.

[21] Michel, O., Sancy, P. and Mondada, H., “KhepOnTheWeb: An Experimental demonstrator in telerobotic and virtual reality,” in *International Conference in Virtual Systems and Multimedia*, 1997.

[22] Siegwart, R. and Goldberg, K, “Robots on the Web,” *IEEE Robotics and Automation Magazine*, 2000.

[23] P. Backes, “Pathfinder Sojourner Rover Simulation Web Page.” [Online]. Available: http://mars.graham.com/wits/