# Experience with Rover Navigation for Lunar-Like Terrains

Reid Simmons, Eric Krotkov, Lonnie Chrisman, Fabio Cozman, Richard Goodwin, Martial Hebert, Lalitesh Katragadda, Sven Koenig, Gita Krishnaswamy, Yoshikazu Shinoda, and William Whittaker

The Robotics's Institute, Carnegie Mellon University
Pittsburgh, PA 15213

Paul Klarer

Sandia National Laboratories
Albuquerque, NM 87185

**Abstract**

*Reliable navigation is critical for a lunar rover, both for autonomous traverses and safeguarded, remote teleoperation. This paper describes an implemented system that has autonomously driven a prototype wheeled lunar rover over a kilometer in natural, outdoor terrain. The navigation system uses stereo terrain maps to perform local obstacle avoidance, and arbitrates steering recommendations from both the user and the rover. The paper describes the system architecture, each of the major components, and the experimental results to date.*

# Introduction

The lure of the Moon is strong — and humans are once again responding to the challenge. One promising, near-term scenario is to land a pair of rovers on the Moon, and to engage in a multi-year, 1000 kilometer traverse of historic sights, including Apollo 11, Surveyor 5, Ranger 8, Apollo 17 and Lunokhod 2 [6]. In this scenario, the rovers would be operated in either autonomous or safeguarded supervisory control modes, and would transmit continuous live video of their surroundings to operators on Earth.

While the hardware aspects of such a mission are daunting — power, thermal, communications, mechanical and electrical reliability, etc. — the software control aspects are equally challenging. In particular, the rover needs capabilities to enable driving over varied terrain and to safeguard its operation. Previous experience with planetary robots (in particular, Lunokhod 2 and the arm on Viking) illustrated how laborious and unpredictable time-delayed teleoperation is for remote operators. A better mode of operation is supervised teleoperation, or even autonomous operation, in which the rover itself is responsible for making many of the decisions necessary to maintain progress and safety.

We have begun a program to develop and demonstrate technologies to enable remote, safeguarded teleoperation and autonomous driving in lunar-like environments. In particular, we are investigating techniques for stereo vision, local obstacle avoidance, position estimation, and mixed-mode operation (autonomous, semi-autonomous, and teleoperated). The aim is to provide both the technologies and evaluations of their effectiveness, in order to enable mission planners to make informed cost/benefit tradeoffs in deciding how to control the lunar rovers.

The research reported here is a descendant of our previous work in rugged terrain navigation for legged rovers [15, 16]. Other efforts have taken similar approaches to navigation for wheeled planetary rovers [1, 3, 4, 5, 17], including the use of obstacle avoidance using stereo vision. Our work is distinguished by its emphasis on long-distance traversal, mixed mode driving, and use of efficient stereo vision using only general-purpose processors.

To date, we have concentrated on the autonomous navigation techniques, and have demonstrated a system that uses stereo vision to drive a prototype lunar rover over a kilometer of outdoor, natural terrain. To our knowledge, this is a record distance for autonomous cross-country driving of a vehicle using stereo and only general-purpose processors.

The issue of operating modes is important in order to reduce the probability of operator fatigue and errors that could be damaging to the rover. We are investigating issues of mixed-mode operation, where a human operator and an autonomous system each provide "advice" on how to drive, with the recommendations arbitrated to produce the actual steering commands to the rover. The idea is to provide a more flexible mode of user interaction, one that utilizes the strengths of both human and machine.

We have also focused on the problem of estimating the rover's position [10]. This is important for any mission, but particularly one that involves long-distance navigation to given sites. Our research on position estimation techniques has concentrated on understanding and filtering sensors such as gyros and inclinometers, using sensor fusion to reduce uncertainty, and developing some novel techniques that involve skyline navigation and Sun tracking [2].

The lessons learned to date are informing our next round of development and experimentation. We are

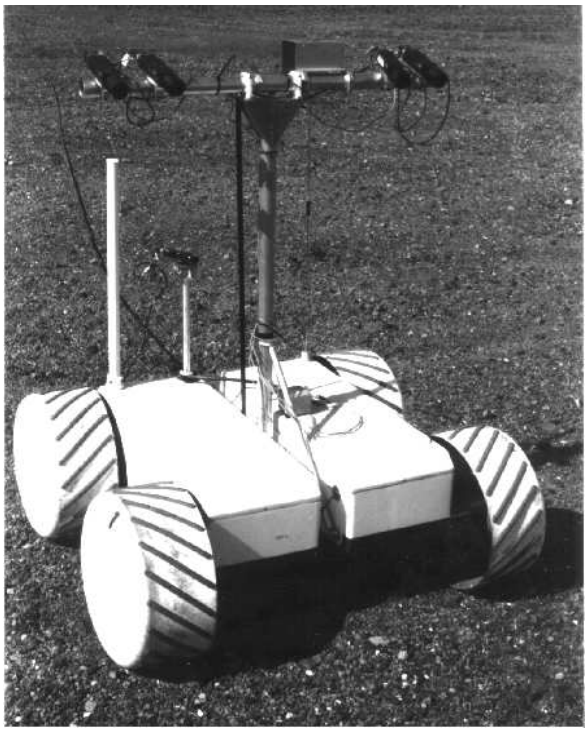

**Figure 1: The Ratler Rover**

working to demonstrate multi-kilometer autonomous traverses and safeguarded teleoperation of up to 10 km, while increasing the complexity of the terrain traversed and the amount of time delay introduced in operating the rover.

The next section describes the rover that is currently being used for our experiments. We then describe the software system developed to drive the rover, and our experimental results. Finally, we address work that is still needed to enable a return to the Moon in this millennium.

# The Ratler

While we are currently designing a new lunar rover [7], we are using a vehicle designed and built by Sandia National Laboratories [11] as a testbed to develop the remote driving techniques needed for a lunar mission. The Ratler (Robotic All-Terrain Lunar Exploration Rover) is a battery-powered, four-wheeled, skid-steered vehicle, about 1.2 meters long and wide, with 50 cm diameter wheels (Figure 1). The Ratler is articulated, with a passive axle between the left and right body segments. This articulation enables all four wheels to maintain ground contact even when crossing uneven terrain, which increases the Ratler's ability to surmount terrain obstacles. The body and wheels are made of a composite material that provides a good strength-to-weight ratio.

Sensors on the Ratler include wheel encoders, turn-rate gyro, a compass, a roll inclinometer, and two pitch inclinometers (one for each body segment). There is a color

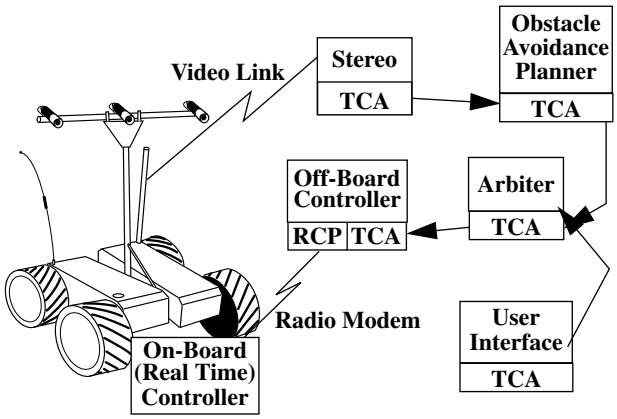


**Figure 2: The Navigation System**

camera for teleoperation, and we have added a camera mast and four black-and-white cameras for stereo vision (only two of which are currently being used). On-board computation is provided by a 286 and a 486 CPU board, connected by an STD bus, which also contains A/D boards and digitizer boards for the stereo cameras.

# The Navigation System

Figure 2 presents a block diagram of the overall navigation software system. Due to power limitations, and for ease of development and debugging, the system is currently divided into on-board and off-board components, communicating via two radio links for video (2.3 GHz) and data (4800 baud). The on-board computers handle servo control of the motors and sensor data acquisition. The other components are run off board, on two Sun workstations (SPARC 10's at 11 MFLOPS).

Each component is a separate process, communicating via message passing protocols. The on-board and off-board controllers communicate over a serial link using the RCP protocol developed at Sandia. The rest of the components communicate via the Task Control Architecture (TCA). TCA is a general-purpose architecture for mobile robots that provides support for distributed communication over the Ethernet, task decomposition and sequencing, resource management, execution monitoring, and error recovery [12]. TCA connects processes, routes messages, and coordinates overall control and data flow.

The basic data flow is that the stereo component produces terrain elevation maps and passes them to the obstacle avoidance planner, which uses them to evaluate the efficacy of traveling along different paths. The arbiter merges these recommendations with the desires of a human operator to choose the best path to traverse (in autonomous mode, there is no operator input). The arbiter then forwards steering and velocity commands to the off-board controller, which ships them to the on-board controller, which then

executes the commands and returns status and sensor information.

While it is convenient to describe the data flow sequentially, in fact all processes operate concurrently. For example, while the obstacle avoidance planner is using one stereo elevation map to evaluate paths, the stereo system is processing another image. Likewise, the arbiter is getting asynchronous path evaluations from the planner and user interface, combining the most recent information to produce steering commands. While it is admittedly more difficult to develop and debug distributed, concurrent systems, they have great advantages in terms of real-time performance and modularity in design and implementation.

## Controller

The on-board controller accepts velocity commands for the left and right pairs of wheels. It uses feedback from the wheel encoders to maintain the commanded velocity over a wide range of terrain conditions. The on-board controller also reports the various sensor readings (compass, gyro, inclinometers, encoders). It expects a "heart-beat" message from the off-board controller, and will halt all motions if not received periodically.

The off-board controller accepts desired steering and velocity commands, and converts these to wheel velocities for the on-board controller. It provides for several safety mechanisms, such as stopping the rover if roll or pitch inclinometers exceed certain thresholds or if it does not receive a new command before the Ratler has traveled a specified distance.

The controller also merges the sensor readings to estimate the position and orientation of the rover. In particular, extensive filtering and screening is performed on the data to reduce noise and eliminate outliers. For example, the compass signal is corrupted by random noise. Based on a spectral analysis of the data, which revealed a cut-off frequency of 0.25 Hz, we implemented several low-pass filters (Butterworth and Bessel). These are effective in suppressing the noise, although they also introduce a 2-3 cycle delay between the filtered value and the signal.

## Stereo

The stereo component used by Ratler takes its input from two black-and-white CCD cameras with auto-iris, 8 mm lenses, mounted on a motion-averaging mast. Its output is sets of (x,y,z) triples, given in the camera coordinate frame, along with the pose of the robot at the time the images were acquired. Using the pose, the (x,y,z) values are transformed into world coordinates to form a (non-uniformly distributed) terrain elevation map. To speed up overall system cycle time, stereo can be requested

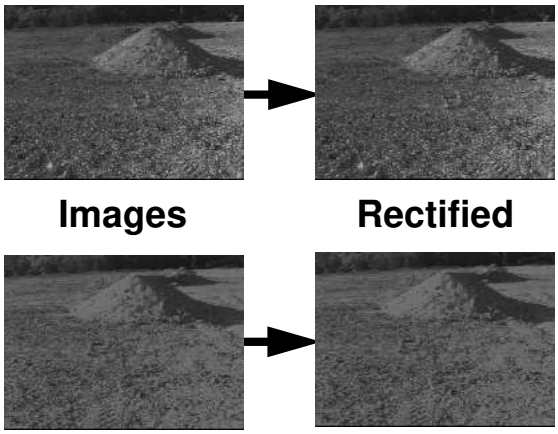

**Figure 3: Rectified Stereo Images**

to process only part of the image, and that at reduced resolution (skipping rows and columns in the image).

The stereo images are first rectified (Figure 3) to ensure that the scan lines of the image are the epipolar lines [12]. The best disparity match within a given window is then computed using a normalized correlation. Disparity resolution is increased by interpolating the correlation values of the two closest disparities. The normalized correlation method is relatively robust with respect to differences in exposure between the two images, and can be used to produce confidence measures in the disparity values.

Care must be taken to ensure that outlier values (caused by false stereo matches) are minimized. Several methods are used to achieve the level of reliability required for navigation. One method eliminates low-textured areas using lower bounds on the acceptable correlation values and variance in pixel intensity. Another method eliminates ambiguous matches (caused by occlusion boundaries or repetitive patterns) by rejecting matches that are not significantly better than other potential matches. Finally, the values are smoothed to reduce the effect of noise. All these methods help to produce elevation maps that accurately reflect the actual surrounding terrain, with only a few centimeters of error.

## Obstacle Avoidance Planner

To decide where it is safe to drive, we have adapted techniques developed in ARPA's Unmanned Ground Vehicle (UGV) program for cross-country navigation [8]. The basic idea is to evaluate the hazards along a discrete number of paths (corresponding to a set of steering commands) that the rover could possibly follow in the next few seconds of travel. The evaluation produces a set of "votes" for each path/steering angle, including "vetoes" for paths that are deemed too hazardous. In this way, the rover

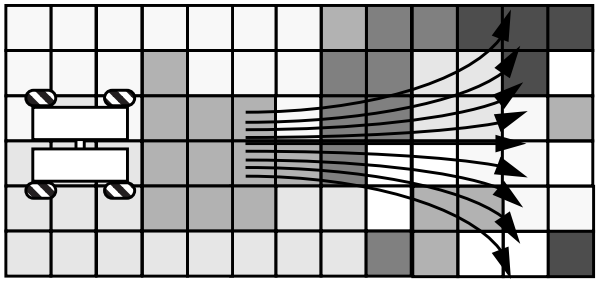

**Figure 4: Evaluating Potential Steering Directions**

steers itself away from obstacles, such as craters or mounds, that it cannot cross or surmount.

The obstacle avoidance planner first merges individual elevation maps produced by the stereo system to produce a 25 cm resolution grid map up to seven meters in front of the rover. Map merging is necessary because the limited fields of view of the cameras do not allow a single image to view sufficient terrain. Currently, we use a rather simple approach that transforms the new map based on the average deviation of elevations between the new and old maps.

To make the stereo computation tractable, a small segment of the stereo image is requested, at reduced resolution. Experiments show that only about 2% of the image is needed for reliably detecting features on the order of 20 cm high. The planner dynamically chooses which portion of the image that the stereo system should process, based on the current vehicle speed, stopping distance, and expected cycle time of the perception/planning/control loop. Typically, stereo is asked for points lying from 4 and 7meters in front of the rover, at a 10 cm resolution.

To evaluate the potential steering commands, the planner uses a detailed model of the vehicle's kinematics and dynamics to project forward in time the expected path of the rover on the terrain. This produces a set of paths, one for each potential steering direction (Figure 4). The planner then evaluates, at each point along the path, the elevations underneath the wheels, from which it computes the rover's roll and the pitch of each body segment. The overall merit of a path depends on the maximum roll or pitches along the path, together with how known is the underlying terrain. The path evaluations are then sent to the arbiter module (paths whose values exceed a threshold are vetoed), along with the pose of the robot at the time of the evaluation.

## Arbiter

The arbiter accepts path evaluations from other components and chooses the best steering angle based on those evaluations. This provides a straightforward way to incorporate information from various sources (such as the obstacle avoidance planner, user interface, route planner) in a modular and asynchronous fashion [13].

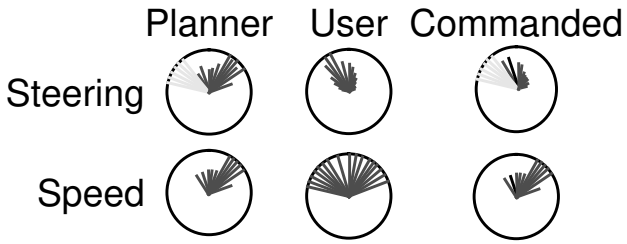


**Figure 5: Arbitrating User & Planner Commands**

Each path evaluation consists of a steering angle, value, and speed (Figure 5). If the value is "veto" (lightly shaded in the figure) then the arbiter eliminates that steering angle from consideration. Otherwise, it combines the recommendations from all sources using a weighted sum. Rather than choosing the arc with the largest value, the arbiter finds the largest contiguous set of steering angles whose values are all within 90% of the maximum value, and chooses the midpoint of that set as the commanded steering angle (for safety, the speed chosen is the minimum recommended speed). The idea is to prefer wide, easily traversable areas over directions that might be a bit more traversable, but have less leeway for error if the rover fails to track the path precisely. We have found this to be very important in practice, as the robot's dead reckoning and path tracking ability are only fair, at best [9].

The path evaluations sent to the arbiter are also tagged with a robot pose. If the tagged pose differs significantly from the rover's current pose, then those path evaluations are ignored. If the evaluations from all the processes are invalidated in this way, then the arbiter issues a command to halt the rover. In this way, the arbiter safeguards against other modules crashing, or otherwise failing to provide timely inputs.

When operating in autonomous mode, the obstacle avoidance planner occasionally cannot find any acceptable path. This typically occurs when the stereo data is noisy or when the robot turns and finds itself facing an unexpected obstacle. To handle such situations, if the arbiter receives several consecutive path evaluations that are all "vetoed," it will command the rover to turn in place by fifteen degrees. This behavior continues until the planner starts sending valid path evaluations again.

## User Interface

Our user interface work has focused on facilitating mixed-mode operation, where the human and rover share responsibility for controlling the robot. The current graphical user interface consists of an "electronic joystick," which utilizes the computer mouse to command the robot's direction and speed, and a number of textual and graphical indicators of pertinent information, such as commanded

and instantaneous robot speeds, roll and pitches, position, and status. Visualization of the terrain is provide by a color camera mounted toward the rear of the Ratler, which is transmitted to a monitor over the microwave radio link.

The user interface supports several driving modes. In the direct teleoperation mode, the human has full control over the rover — almost all safeguarding is turned off. Direct teleoperation is necessary when the rover gets into situations where the software would otherwise prevent motion. For instance, there may be occasions where the pitch limits must temporarily be exceeded to drive the rover out of a crater. This mode would be reserved for experienced drivers in exceptional situations.

On the other side of the spectrum, in the autonomous mode the software system has complete control over the robot, choosing the direction to travel based on the stereo maps. While the majority of our experiments have consisted in letting the robot "wander" autonomously, while avoiding obstacles, we have done some experiments that use a simple planner to add goal-directed input to the arbiter, to bias the robot in a particular direction.

The third mode, safeguarded teleoperation, is seen as the standard way in which the lunar rover will be operated. In this mode, input from the human and the obstacle avoidance planner are combined: the user presents a desired direction to travel, and the obstacle avoidance planner can veto it, causing the robot to refuse to travel in that direction. The idea is that the software safeguards should prevent the user from damaging the rover, but not otherwise interfere with the control. This mode is implemented by placing a Gaussian distribution over the user's arcs, centered at the desired heading (Figure 5), and having the arbiter combine the user and planner inputs. If the human chooses not to provide input, the arbiter just considers the planner's inputs. In this way, operator fatigue can be reduced by letting the robot operate on its own when it is in benign terrain, while still enabling the user to take over control at any moment.

# Experimental Results

We have done extensive testing of the system, concentrating on autonomous navigation. Most of the experiments were performed at a slag heap in Pittsburgh (Figure 6), on an undulating plateau featuring some sheer cliffs and sculpted features (mounds and ridges).

Early experiments tested the dead reckoning capability of the system and its ability to avoid discrete obstacles. After characterizing the response of the vehicle and improving the position estimation [9, 10], the rover was able to navigate hundreds of meters with minimal human intervention. To date, our longest contiguous run has been 1,078 m, where 94% of the distance was traversed in autonomous mode and the rest in direct teleoperation mode. Average speed attained for that run was about 10 cm/s, although we have driven, with similar results, at over twice that speed. In all, over 3000 stereo pairs were processed and used by the planner for path evaluations.

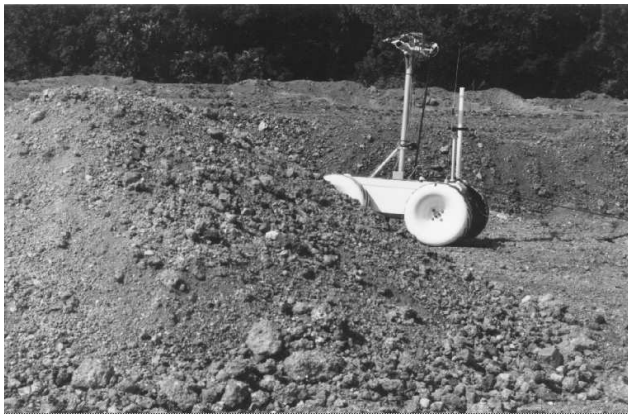

**Figure 6: The Ratler at the Pittsburgh Slag**

The cycle time for stereo is about 1 second on a SPARC 10, and cycle time for the obstacle avoidance planner is 0.5 second (other computation times are minimal). Since the largest latency is in acquiring and transmitting image pairs (about 2 seconds), we are investigating using wireless Ethernet to speed this up. The overall cycle time, in which perception and planning is concurrent, is about 3 seconds.

In other experiments, we tested the ability of the arbiter to combine inputs from multiple sources: the obstacle avoidance planner and either the user or a path-tracking module. These experiences pointed out the need to improve the arbitration model to choose more intuitive steering commands (such as by biasing toward directions that have recently been chosen). This is particularly important when dealing with human operators, who need a fairly predictable model of the system behavior in order to feel comfortable and confident about interacting with it.

The experiments also pointed out the need to increase the stereo field of view (by using two pairs of cameras), make stereo and map merging more robust to sensor uncertainty, request stereo data points more judiciously, and improve dead reckoning accuracy.

# Ongoing and Future Work

To achieve the ambitious goals of the mission scenario (1000 km traverse over two years), we need to harden and extend our techniques. Certain changes, such as those described in the previous section, merely involve incremental improvements. Other needed extensions are of a more fundamental nature. One involves the addition of

short-range proximity and tactile sensors to provide more reliable safeguarding. We have recently acquired a laser range sensor that can provide coverage in front of the rover at ranges of one to two meters, and are developing sensor interpretation algorithms to detect impending hazards, especially cliffs and other drop-offs. We are also considering tactile sensors to detect high-centering collisions with the underside of the rover.

We also plan to add more extensive internal monitoring of the robot's health and integrity. For example, we will monitor battery voltage and motor currents, and autonomously "safe" the vehicle if they exceed given thresholds.

Our experimental work will take two directions: we will continue to demonstrate and quantify autonomous navigation capabilities using stereo, and we will investigate more carefully issues of mixed-mode and safeguarded teleoperation. This includes quantifying the performance improvements gained by adding various technologies, such as stereo-based driving, use of high-level command inputs, etc. Our next major goal is a 10 km safeguarded teleoperated traverse in rougher, more lunar-like terrain.

# Conclusions

This paper has presented a system for autonomously and semi-autonomously driving a prototype lunar rover in natural, outdoor terrain. The navigation system uses a combination of on-board and off-board computation to control the vehicle, process stereo images, plan to avoid obstacles, and integrate machine and human recommendations regarding the travel direction. Although the system uses mainly classical sensors and known algorithms, it has achieved unprecedented results, enabling long-distance (greater than 1 km) outdoor traverses. The key contributions are in tailoring the general ideas to a specific robot performing a specific task, and in demonstrating practical and unprecedented performance.

Our experiments have demonstrated basic competence in driving to avoid cliffs and mounds, but much more work needs to be done in order to produce a system that can behave reliably over many weeks and kilometers. In particular, we have targeted the areas of safeguarding and remote teleoperation as worthy of further investigation.

It is important to realize that safeguarding and autonomous navigation can have profound impact on the ease and reliability of remote driving of a lunar rover. On the other hand, such systems admittedly add complexity to the hardware and software requirements of a rover. We need to perform careful experiments to quantify the value added by these technologies, in order to demonstrate their effectiveness for near-term lunar missions.

# Acknowledgments

This research was partly sponsored by NASA, under grants NAGW-3863 and NAGW-1175. We gratefully acknowledge assistance from Ben Brown, Michel Buffa, Yasutake Fuke, Luc Robert, and Wendy Amai.

# References

[1] R. Chatila, R. Alami, *et al*. Planet Exploration by Robots: From Mission Planning to Autonomous Navigation. In *Proc. Intl. Conf. on Advanced Robotics*, Tokyo, Japan, Nov. 1993.

[2] F. Cozman and E. Krotkov. Mobile Robot Localization using a Computer Vision Sextant. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, Nagoya, Japan, May 1995.

[3] J. Garvey, A Russian-American Planetary Rover Initiative, AIAA 93-4088, Huntsville AL, Sept. 1993.

[4] E. Gat, R. Desai, *et al.* Behavior Control for Robotic Exploration of Planetary Surfaces. *IEEE Transactions on Robotics and Automation*, **10:4**, Aug. 1994.

[5] B. Hotz, Z. Zhang and P. Fua. Incremental Construction of Local DEM for an Autonomous Planetary Rover. In *Proc. Workshop on Computer Vision for Space Applications,* Antibes France, Sept. 1993.

[6] L. Katragadda, *et al*. Lunar Rover Initiative — Preliminary Configuration Document, Tech. Report CMU-RI-TR-94-09, Carnegie Mellon University, 1994.

[7] L. Katragadda, J. Murphy and W. Whittaker. Rover Configuration for Entertainment-Based Lunar Excursion. In *Intl. Lunar Exploration Conference*, San Diego, CA, Nov. 1994.

[8] A. Kelly. A Partial Analysis of the High Speed Autonomous Navigation Problem. Tech Report CMU-RI-TR-94-16. Robotics Institute, Carnegie Mellon University, 1994.

[9] E. Krotkov and M. Hebert, Mapping and Positioning for a Prototype Lunar Rover. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, Nagoya, Japan, May 1995.

[10] E. Krotkov, M. Hebert, M. Buffa, F. Cozman and L. Robert. Stereo Driving and Position Estimation for Autonomous Planetary Rovers. In *Proc. IARP Workshop on Robotics In Space*, Montreal, Canada, July 1994.

[11] J. Purvis and P. Klarer. RATLER: Robotic All Terrain Lunar Exploration Rover. In *Proc. Sixth Annual Space Operations, Applications and Research Symposium*, Johnson Space Center, Houston TX, 1992.

[12] L. Robert, M. Buffa and M. Hebert. Weakly-Calibrated Stereo Perception for Rover Navigation. In *Proc. Image Understanding Workshop*, 1994.

[13] J. Rosenblatt, DAMN: A Distributed Architecture for Mobile Navigation, In *AAAI Spring Symposium on Software Architectures for Physical Agents,* Stanford CA, March 1995.

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

[15] R. Simmons, E. Krotkov, W. Whittaker, *et. al.* Progress Towards Robotic Exploration of Extreme Terrain. *Journal of Applied Intelligence,* **2**, 163-180, 1992.

[16] D. Wettergreen, H. Pangels and J. Bares. Gait Execution for the Dante II Walking Robot. In *Proc. IEEE Conference on Intelligent Robots and Systems*, Pittsburgh PA, August 1995.

[17] B. Wilcox, L. Matthies, *et al.* Robotic Vehicles for Planetary Exploration. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, Nice France, May 1992.