Stereo Vision Based Navigation for Sun-Synchronous Exploration

Chris P. Urmson, M. Bernardine Dias, Reid G. Simmons

The Robotics Institute, Carnegie Mellon University, Pittsburgh, USA
{curmson, mbdias, reids}@ri.cmu.edu

Abstract

This paper describes the navigation system used on a prototype sun-synchronous robot. Sun-synchrony is a concept that will enable exploration missions by solar-powered rovers that could last months or years. This paper presents navigation algorithms developed for traversing natural terrain robustly. The novel elements of this work are the refinements necessary to transform laboratory-demonstrated technologies into a form useful for robust, sun-synchronous exploration. Results of a field experiment in the Canadian Arctic, where the robot traversed 6.1km, 90% autonomously, are also presented.

1.1 Introduction

Sun-synchronous exploration is an energy-cognizant strategy for planetary exploration [10][18] that utilizes knowledge of terrain, time and rover characteristics to maximize solar energy gathered while performing a mission. This concept will enable exploration missions by solar powered rovers that could last months or years. Hyperion (figure 1), a solar powered robot, was developed to demonstrate this idea on Earth. In July 2001 Hyperion was deployed to Devon Island, above the Arctic Circle, to perform experiments in sun-synchrony. The culmination of these experiments was a 24-hour sun-synchronous route, which Hyperion planned and executed [19]. This paper describes the obstacle detection and navigation systems utilized on Hyperion and presents relevant results of the field experiments.

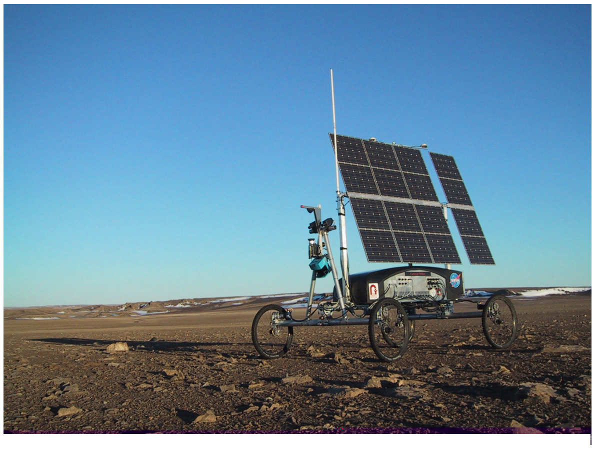

Figure 1: *Hyperion at the field experiment site.*

1.2 Navigation Problem

Hyperion's navigation problem can be divided into two parts: (1) generation of a sun-synchronous route and (2) execution of the route. A mission planner generates routes from digital elevation maps and knowledge of the motion of the sun[15]. The goal of the mission planner is to output routes that maximize solar input to Hyperion's fixed solar panel while completing a feasible circuit of a desired radius. The mission planner is only capable of planning for features that are detectable at the resolution of the elevation map (typically 25m or greater), such as hills and valleys.

To complement the mission planner, a finer resolution motion planner (navigator) is required to move the robot around obstacles that are too small to appear in the elevation maps (or are unknown at the time of mission specification) while achieving waypoints determined by the mission planner. The navigator is the focus of this paper.

Important to the design and implementation of a navigator are the characteristics of the environment. The high arctic terrain that Hyperion is designed for is typical of many regions of the Moon and Mars. The terrain is mostly rolling with discrete large boulders and impassable rocks interspersed at a low density. A particular focus of this paper is the refinements to laboratory navigation algorithms that were made to allow for a successful demonstration of long duration, robust, sun-synchronous exploration.

2 Related Work

A majority of work in robot navigation has dealt with robots operating on a flat plane with discrete obstacles[6]. In recent years, there has been increased interest in algorithms capable of driving robots over rough terrain. Much of this work also maintains the assumption that space can be easily divided into what is and what is not traversable [1][5][14]. Natural terrain rarely provides this simple distinction. There is generally a gradation between terrain that is easily traversed and that which is completely impassable. Without modeling this variability, a robot will be paralyzed due to excessive caution or will take unnecessary risks. The abrupt boundary between traversable and impassable can also lead to potentially dangerous instabilities in the navigation algorithm when the terrain is viewed from slightly different perspectives.

Researchers are beginning to explore descriptions of terrain that better encode its variability. Continuous measures of terrain have been implemented as fuzzy sets [7][9]. These representations are generally combined with fuzzy logic controllers, which suffer local minima problems similar to potential field approaches [6]. Other work describes traversability as a continuous value based on the weighted combination of terrain metrics [2][3][11]. Generally, systems that use obstacle detection incorporate it as a reactive behavior with little global planning capability [8]. By combining a path planner with a local obstacle detection system useful behavior can be generated in unknown or partially known environments.

The novel elements of this work are the refinements necessary to transform laboratory-demonstrated technologies into a form useful for robust, sun-synchronous exploration. Robustness is essential for sun-synchrony due to the extended duration of the missions. Making local navigation decisions that increase the sun-pointing of the solar panels while avoiding obstacles is a second important area for sun-synchrony. The culminating field experiment with Hyperion provided a unique opportunity to test and verify these improvements.

3 Design Overview

To maximize robustness, an early design decision was to allow teleoperators to resolve pathological navigation problems that were possible, but unlikely to occur. By focusing on the central navigation problems, rather than worrying about low-probability "what if" scenarios, the team was able to ensure that Hyperion's core navigation capabilities worked reliably.

The navigation system can be divided into four principle components:

Navigator: perceive the terrain and select the appropriate action

Vehicle controller: convert actions into hardware commands

Health monitor: monitor the state of the vehicle and disengage the autonomy system and notify operator in case of problems

State Estimator: fuse sensor data to provide the approximate state of the robot

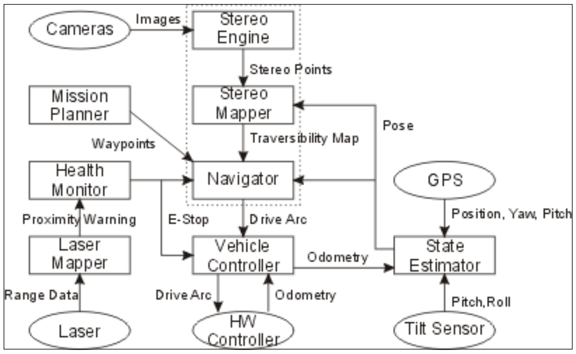

Figure 2: *Hyperion's software architecture.*

A teleoperation interface allows remote operators to assist the robot when unusual situations arise. The architecture illustrated in figure 2 allows a human operator to intercede at any point by passing drive arcs directly to the vehicle controller. The navigator however, will continue to evaluate the terrain and monitor the robot's progress; this approach allows the autonomy system to continue moving as soon as the operator cedes control. In practice, this often meant it was impossible for observers to detect the transition between teleoperation and autonomous operation. The balanced combination of human teleoperation and autonomy proved extremely reliable and robust.

4 Perception

Hyperion utilizes a two-tiered obstacle detection system to guarantee safety. A stereo vision system provides a detailed evaluation of the nearby terrain while a laser range finder monitors the ground immediately in front of the robot. The laser range finder is configured as a "virtual bumper"; it prevents the robot from hitting any undetected obstacles by causing the health monitor to issue a stop command when an imminent collision is detected. The teleoperator can then assess and correct the situation before handing control back over to the autonomous navigator.

A common failure point for many navigation systems are stereo induced errors. Low visual texture can blind stereo systems, providing no-information for the navigation system to operate on. In general, it is prudent to consider unsensed terrain unsafe, but given the nature of the terrain (sparse obstacles, softly rolling) and that a safety sensor (the virtual bumper) is in place, considering the terrain optimistically makes sense. This encourages the use of a navigation system that allows terrain that is unsensed by the stereo vision system to be traversed. This approach helps limit the number of false obstacles that often cause great inefficiency for stereo based navigation algorithms. However, even with the optimistic assumption, it is important to perform the stereo analysis of the terrain as accurately as possible to allow Hyperion to navigate efficiently.

4.1 Stereo Filtering

Point errors due to miscorrelations are a common problem in stereo vision generated 3D data. A common approach to preventing miscorrelation is to filter the resultant disparity image by rejecting regions where the input image doesn't have much visual texture. A similar technique is to filter pixels when the correlation is not sufficiently unique. These techniques often remove valid pixels because they cannot determine if a low probability match is actually a true reading. This can cause a point cloud to contain sparse data in regions of the image with low visual texture. Given domain knowledge, and the safety provided by the virtual bumper, a more optimistic filtering technique can be used.

The world Hyperion navigates in can be modeled as a (relatively) smoothly varying 2 ½ D surface. Any large step or spike in the disparity data can be assumed to be noise and should be discarded. This approach allows the algorithm to utilize regions of low visual texture while still providing useful data. To check for steps in the disparity image, each pixel is compared with its neighbors. If its disparity is more than some threshold value different than any of its neighbors, it is discarded. This algorithm can be easily and efficiently implemented. Small patches of bad data can be removed by this approach but the skeleton of large patches will remain.

A final filtering step compares the computed height of each pixel to the expected ground plane. If it is too far from the ground plane, the pixel is discarded. The combination of these two approaches dramatically reduced the number of false obstacles detected by the stereo mapping system while maintaining a dense point cloud.

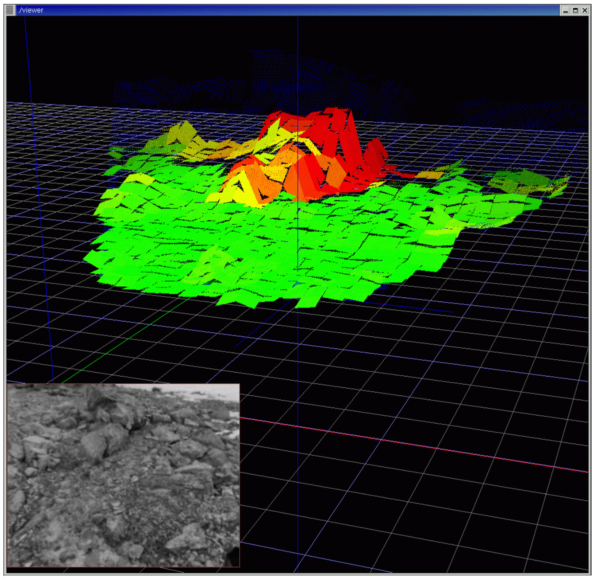

Figure 3: *3D view of terrain rendered by an analysis tool.*

4.2 Traversability Analysis

Hyperion's obstacle detection algorithm is derived from MORPHIN[11]. For each stereo image, the algorithm generates a traversability map of the local terrain. To do this, the local terrain is divided into cells. Groups of cells are combined into overlapping robot-sized patches. For each patch the algorithm finds the best plane that represents the perceived terrain.

Traversability is determined by looking at three metrics: the slope, roughness and "step height". Each metric is normalized such that they can be compared directly. The traversability of the patch is determined by the worst of these three values.

Regions of the stereo footprint that contain more points generate a better estimate of the actual terrain. To capture this, a certainty value is computed for each patch. Certainty is calculated as a function of the number of points in a patch and the evenness of the distribution of the points over the patch.

Critical to the success of the obstacle detection algorithm was the development of tools that allow in-depth visualization of its inner workings. The tool illustrated in Figure 3, allows the user to directly compare the values generated by the statistical analysis with the geometry of the terrain (and point cloud). This capability provides insight into how parameters (such as the amount of "roughness" that is considered an obstacle) should be adjusted to generate correct behavior.

5 Navigation

The navigator uses local terrain evaluations to plan traversable paths to goals generated by the mission planning software or a human operator. The navigator builds maps of the local terrain on the scale of 100m x 100m.

Path planning to a goal is performed using an enhanced version of the D* algorithm [12][13]. This modified algorithm calculates the path cost as a combination of

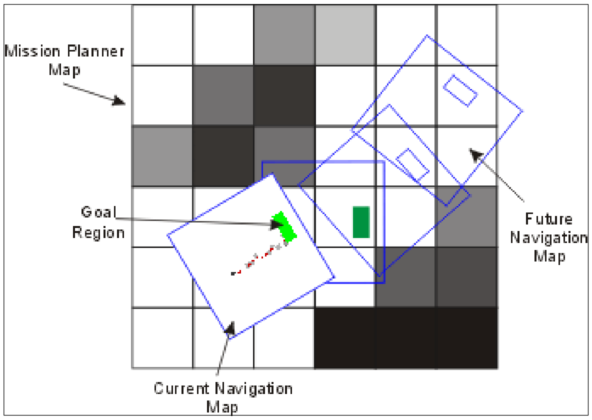

Figure 4: *An illustration of the navigation map orientation.*

the cost to traverse an arc and the cost from the end of an arc to the goal. The basic procedure of the navigator when evaluating paths is as follows:

- Update the robot position via the state estimator
- Obtain stereo information from the stereo mapper
- Compute the traversability analysis
- Insert the new traversability data into the D* map
- Update the position to account for movement of the robot during stereo data evaluation
- Evaluate the cost along each of the arcs (total sum of cost along the arc plus cost from the end of the arc to the nearest point in the goal region), vetoing any arc that collides with an obstacle
- Choose the arc that results in the lowest cost, if several arcs with equal cost are found, one is selected arbitrarily
- Send radius, speed, and safety timeout for this arc to the controller
- If the goal is not reached, repeat procedure (Thus, the robot travels a fraction of the chosen arc, and then the entire procedure is repeated)

A combination of map orientation and goal regions are used to help maximize Hyperion's sun-pointing while navigating between waypoints and around obstacles. Each navigation map is oriented such that the principal axis is aligned along the path between the current goal and the next goal, as illustrated in figure 4. This direction is the direction of travel the mission planner believes will maximize solar power gain.

Waypoints, generated by the mission planner, are specified as goal regions whose dimensions and orientation are determined at runtime. By aligning goal regions perpendicular to the preferred direction of travel, the 8-connectedness of the map encourages the navigator to generate paths that maximize sun-pointing.

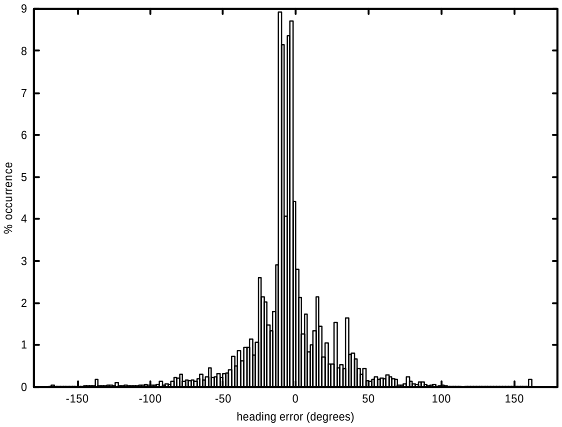

Figure 5. *The distribution of heading error.*

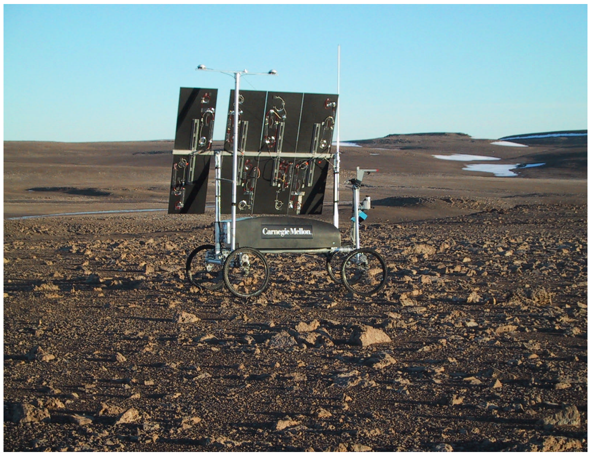

Figure 6: *Hyperion in a rock field with a mixture of traversable and intraversable obstacles.*

6 Results

To evaluate the concept of sun-synchronous navigation, Hyperion was deployed to Haughton Crater (on Devon Island), in Arctic Canada. As part of an end-to-end system test, the mission planner generated a sequence of waypoints marking out a 5.8km sun-synchronous circuit that the navigator was to execute over a period of 24 hrs while human operators monitored progress. For recharging cycles between waypoints, operators would orient the robot as required by the mission planner. Further information regarding this experiment can be found in [15][19].

6.1 Terrain Evaluation

The filtered stereo mapping data generated by the stereo algorithm had planar errors ranging from -5cm to +20cm increasing with distance from the cameras. In practice, this error did not affect the validity of maps generated by the stereo obstacle detection algorithm. At range, it is sufficient to know approximately where an obstacle is located, so that gross motions can be performed to avoid it. As the robot travels towards obstacles, the position of the obstacle is revised repeatedly, providing an accurate location when the robot most needs it. In practice the stereo based obstacle detection is more reliable than the "virtual bumper" which generates a number of false positives due to an algorithm that is overly simplistic for the arctic terrain.

6.2 Robustness

The majority of Hyperion's traverse was performed autonomously. Post analysis of mission telemetry revealed that Hyperion drove 6.1 km, operating autonomously for 90% of the time. At no point was it necessary for human observers to physically interact with the robot. Tele-operators found that manually adjusting the speed Hyperion used had an impact on Hyperion's ability to navigate through dense obstacle fields. Due to the decoupling of forward speed and steering, Hyperion can change its heading in a much shorter distance when traveling at lower speeds, allowing it to navigate tight constraints that it would not be capable of at higher speeds.

The navigator and obstacle detection algorithms proved very reliable. The level of performance was evident in the navigator's ability to traverse cluttered terrain without hitting any rocks that it could not drive over.

6.3 Efficiency

A reasonable metric for efficiency of the navigator is heading error: the difference between the optimal heading and the actual heading. Over the 24 hour traverse, the robot had an average heading error of 5.4°, the average of the absolute value of the heading error was 21.0°. The distribution of the heading error is shown in Figure 5. In the context of sun-synchrony this meant the robot received 93% of the maximum energy the mission planner expected it to receive.

The average density of rough or impassable terrain Hyperion traversed was 6.9%. Rough terrain, in this context, is any terrain that would significantly disturb Hyperion (a lower bound would be approximately rocks that are 5-8cm in height). During several legs, the density of rough terrain was considerably higher, reaching a maximum of 34.2%. Figure 6 shows Hyperion navigating in a dense rock field while figure 7 shows a navigation map representative of similar difficult terrain.

Another important metric is the amount of steering activity generated by the navigator. In a world with static obstacles, a navigator should change its desired curvature gradually. Rapid changes of the steering are undesirable as they waste energy and cause excessive mechanical wear. On average, during each cycle, Hyperion changed the curvature of the arc it was traveling along by 0.03m^{-1}; or by less than one arc in its discretized arc set. This smoothness implies that the stereo perception system was also stable, since if the stereo system were producing a significant number of false obstacles, the navigator would be forced to make abrupt movements to avoid them. Figure 8 shows the relationship between terrain difficulty and navigation smoothness. In this data, there is a general trend that shows that as the percentage of rough terrain increases, steering activity also increases.

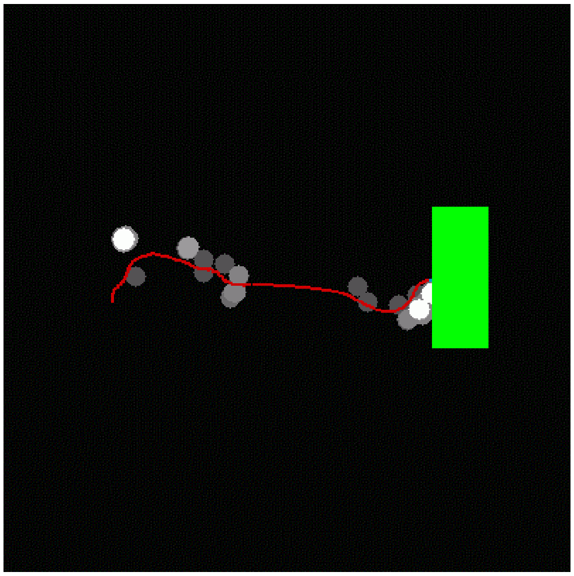

Figure 7: *The navigation map for a traverse where Hyperion crossed a cluttered rock field. Terrain is classified from safe or unknown (black) to unsafe (white). The rectangle indactes the goal region.*

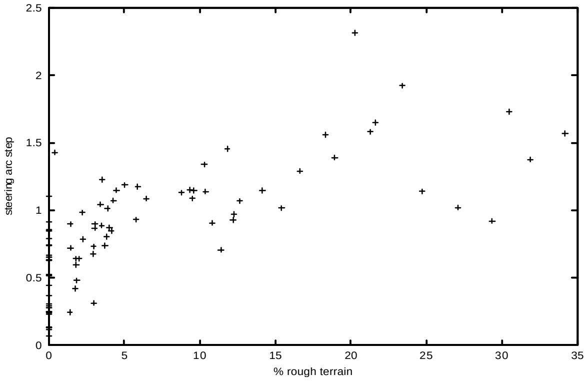

Figure 8: *The relationship between terrain roughness and steering activity. Each point represents a route segment, approximately 70m of travel.*

7 Future work

The exercise of developing the navigation system for Hyperion pointed towards several development areas in which there could be significant payoff.

7.1 Variable Speed Control

Operators can control the velocity at which the navigator drives Hyperion. In tight quarters, slowing the robot improved its ability to navigate around obstacles due to a relative increase in nimbleness. By monitoring the local obstacle density, Hyperion could have autonomously decreased its speed as necessary. This would decrease human input further and would likely result in a more robust robot which does not sacrifice speed unnecessarily.

7.2 General Rough Terrain

Even when utilizing continuous values to describe the traversability of terrain, there are many subtleties about the terrain that are missed. Often the function used to convert sensor data to cost must be pessimistic. A good example is a rock that is short enough to be cleared by the body of the robot, but too large for the rover's wheels to drive over. By considering the worst case contact, our current algorithms would not allow the rover to straddle the rock, even if it is the only possible path. Improving traversability analysis and terrain understanding will become particularly important as mobile robots operate in more difficult regions with greater reliability requirements.

7.3 Generalization of the Algorithm

The NASA Jet Propulsion Laboratory is developing a common object based framework, called CLARAty, to allow robotic software to be more portable between software and hardware platforms [16]. As part of this effort, the navigation software described in this paper will be ported to this framework. This will lead to more robust and generally useful software while providing a framework within which various navigation algorithms can be more directly compared.

8 Conclusions

Hyperion demonstrated robust, sun-synchronous navigation. The development and application of several refinements enabled Hyperion to navigate autonomously for 24 hours, traversing more than 6km, without a navigation failure. The refinements developed for Hyperion are illustrative of how domain knowledge can be applied to increase navigation robustness.

Acknowledgments

This paper describes the work of the Sun-synchronous Navigation project and all of its members are important contributors. The authors acknowledge and thank Dimi Apostolopoulos, Jesse Boley, Stewart Moorehead, Benjamin Shamah, Sanjiv Singh, Tony Stentz, James Teza, Paul Tompkins, Vandi Verma, Michael Wagner and David Wilkinson. Field experimentation was conducted in collaboration with the NASA Haughton-Mars project, Pascal Lee, Principle Investigator. This work is supported by NASA under grant NAG9-1256.

9 References

[1] P. Belluta et al. “Terrain Perception for Demo III”, Proc. IEEE Intelligent Vehicles Symposium, Dearborn, USA, October 2000.

[2] J. Biesiadecki et al. “The Athena SDM Rover: a Testbed for Mars Rover Mobility”, Proc. International Symposium on Artificial Intelligence and Robotics & Automation in Space, St-Hubert, Canada, June 2001.

[3] S. Golberg, M. Maimone & L. Matthies. “Stereo Vision and Rover Navigation Software for Planetary Exploration”, Proc. IEEE Aerospace Conference, Big Sky, USA, March 2002.

[4] A. Kelley and A. Stentz. “Rough Terrain Autonomous Mobility – Part 1: A Theoretical Analysis of Requirements”, Autonomous Robots, No. 5, May 1998, pp 129-161

[5] S. Laubach and J. Burdick. “An Autonomous Sensor-Based Path-Planner for Planetary Microrovers”, Proc. IEEE International Conference on Robotics and Automation, Detroit, USA, May 1999.

[6] J-C. Latombe. “Robot Motion Planning”, Kluwer Academic Publishers, 1991.

[7] A. Martin-Alvarez et al. “Fuzzy Reactive Piloting for Continuous Driving of Long Range Autonomous Planetary Micro-Rovers”, Proc. IEEE Aerospace Conference, 1999.

[8] H. Seraji et al. “Safe Navigation on Hazardous Terrain”, Proc. IEEE International Conference on Robotics and Automation, Seoul, Korea, May 2001.

[9] H. Seraji. “Traversability Index: A new concept for Planetary Rovers”, Proc. IEEE International Conference on Robotics and Automation, Detroit, USA, May 1999.

[10] K. Shillcutt. “Solar Based Navigation for Robotic Explorers,” Ph.D. thesis, CMU-RI-TR-00-25, October 2000.

[11] S. Singh et al. “Recent Progress in Local and Global Traversability for Planetary Rovers”, Proc. IEEE International Conference on Robotics and Automation, San Francisco, USA, April 2000.

[12] A. Stentz. “Optimal and Efficient Path Planning for Partially-Known Environments”, Proceedings of IEEE International Conference on Robotics and Automation, volume 4, pp.3310-3317, 1994.

[13] A. Stentz. “The Focused D* Algorithm for Real-Time Planning”, Proceedings of International Joint Conference on Artificial Intelligence, August 1995.

[14] A. Stentz and M. Hebert. “A Complete Navigation System for Goal Acquisition in Unknown Environments”, Proc. IEEE/RSJ International Conference On Intelligent Robotic Systems, Aug 1995.

[15] P. Tompkins et al. “Mission planning for the Sun-synchronous Navigation Field Experiment”, Proc. IEEE International Conference on Robotics and Automation, Washington, USA, May 2002.

[16] R. Volpe et al. “The CLARAty architecture for robotic autonomy”, Proc. IEEE Aerospace Conference, 2001.

[17] D. Wettergreen et al. “Developing Nomad for Robotic Exploration of the Atacama Desert”, Robotics and Autonomous Systems, February 1999.

[18] D. Wettergreen et al. “Robotic Planetary Exploration by Sun-Synchronous Navigation,” Proc. International Symposium on Artificial Intelligence and Robotics & Automation in Space, St-Hubert, Canada, June 2001.

[19] D. Wettergreen et al. “First Experiment in Sun-Synchronous Navigation”, Proc. IEEE International Conference on Robotics and Automation, Washington, USA, May 2002.