Field Experiments in Rover Navigation via Model-Based Trajectory Generation and Nonholonomic Motion Planning in State Lattices

Mihail Pivtoraiko[1,2], Thomas M. Howard[1], Issa A.D. Nesnas[2], Alonzo Kelly[1]
[1]*Robotics Institute, Carnegie Mellon University*
[2]*Jet Propulsion Laboratory, California Institute of Technology*
{mihail, thoward, alonzo}@ri.cmu.edu, nesnas@jpl.nasa.gov

Abstract

This paper presents field experiments of two novel approaches to local and regional motion planning applied to planetary rover navigation. The first approach solves the two-point boundary value problem using a model-based trajectory optimization technique that inverts an arbitrary dynamics model to generate a feasible motion plan. The second approach utilizes this result to build a special discretization of the state space that allows employing standard search algorithms for solving the motion planning problem. These approaches enable robot autonomy by considering the robot's dynamics, efficiently searching a finely discretized state space, and allowing the reuse of previous planning computation to improve runtime. We present results from the experiments on the Rocky 8 and FIDO planetary rover prototypes in the NASA/JPL Mars Yard.

1. Introduction

Autonomous rover navigation is an enabling technology for space exploration. It offers a number of significant benefits:

- lower cost and effort for mission management
- increased scientific return
- efficient multi-robot coordination for exploration or construction
- improved motion quality when teleoperation is infeasible or impractical

Experience from the recent Mars Exploration Rover (MER) mission shows that significant improvements in rover autonomy, including navigation, are necessary. Severe environmental hazards such as significant wheel slip and dense obstacle fields make autonomy difficult. This makes autonomous rover navigation a challenging problem because the terrain can be arbitrary, dynamics can be challenging, perceptual horizons and computational resources are typically limited, and motion efficiency and energy expenditure must be considered (Figure 1).

Figure 1. Rover operating in rough terrain.

In this paper we present the results of field tests conducted at JPL of two new motion planning algorithms developed at Carnegie Mellon. The algorithms are applied to both local and regional rover navigation problems. They address the above challenges by considering model dynamics during planning and by searching a considerably more expressive space of actions. In this paper, the term dynamics will refer generally to differential equations encoding the mapping from inputs to outputs. We used first order (e.g. kinematic velocity) models rather than second order (e.g. Newton-Euler) models.

1.1. Technical Approach

We are concerned with the problem of nonholonomic motion planning in complex environments. In general, this involves finding a trajectory between a pair of boundary states while satisfying a number of constraints [1]. Typically, these approaches take the form of a continuum optimization or a sequential search process. Continuum optimization provides a continuum solution provided an initial guess is within the radius of convergence of the global optimum. Conversely, sequential search

processes provide a globally optimal motion plan, but generate a solution only if it exists at the chosen resolution.

In this paper, we describe the application of two motion planning algorithms: a model-based trajectory generator and a state lattice planner. The model-based trajectory generator [2] is based on the continuum optimization techniques and minimizes boundary state constraint error by optimizing parameterized control inputs. The state lattice planner [3][4] is a sequential search process executed on a search space that captures the full maneuverability of the rover at a finite resolution. The state lattice planner can be used for regional or global motion planning in complex environments, due to its resistance to the effects of local minima. The model-based trajectory generation algorithm can be applied in local motion planning, path following of state lattice planner solutions, and for generating the lattice primitives used by the state lattice planner.

1.2. Prior Art

Robot navigation in unknown environments has been explored extensively. Some leading approaches [5][6][7][8] evaluate a number of local motion alternatives sampled in control or input space. While these approaches work well in open or simple areas, the limited expressiveness of these search spaces leads to suboptimal behavior in complex or cluttered environments (Figure 2).

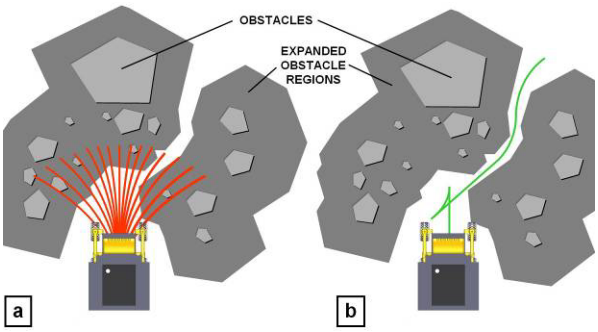

Figure 2. Robot navigation in complex environments. A typical search space is shown in (a), where every sampled trajectory intersects with an obstacle. By searching a better approximation to the entire reachable space of the vehicle, a solution can be found (b).

Such limitations have been addressed in other work by enabling the rover to evaluate several layers of motion alternatives in a tree-like structure. Such structures are evaluated by a graph search to determine the best trajectory, consisting of several concatenated path elements [9]. Our method in [3][4] further extends this approach by framing the path selection process as a problem of constrained motion planning. Thanks to the efficiency of our motion planner, it fits directly into the rover navigation framework.

Motion planning under mobility and environmental constraints has received considerable attention. A standard motion planning method that satisfies global constraints is grid search [1]. Due to its efficiency, it is commonly applied in robot navigation [6][7]. Its efficiency results from its reduced dimension (2D), reduced outdegree (4 or 8 edges per node) and its network (graph) topology which permits the use of efficient search algorithms.

Satisfying local (differential) constraints adds further complexity. Our approach takes the best elements from the grid search approach and extends them to satisfy differential (nonholonomic and continuity) constraints.

2. Model-Based Trajectory Generation

The model-based trajectory generation algorithm in [2] is an optimization method that minimizes boundary state constraint error by modifying parameterized control inputs (Figure 3). The controls are parameterized representations in the form of low-order polynomial spiral or spline primitives in order to minimize the degrees of freedom in the system and to achieve smooth trajectories between boundary states. The correction factor for the optimization is determined numerically through forward or central differences of simulations of the motion model in response to perturbing parameterized freedoms.

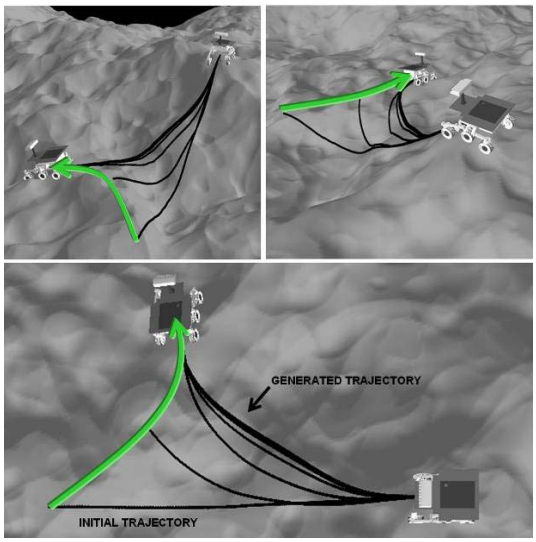

Figure 3. Trajectory optimization considering terrain shape. The model-based trajectory generator optimizes the response to the control inputs to minimize the boundary state constraint error.

This approach is highly favorable for planetary rover applications because of the real-time performance of the algorithm and the flexibility of the motion model. The model dynamics included in the motion model can be scaled in accordance with the difficulty of the environment (e.g. wheel slip, sliding, terrain shape), computational resources available on the platform, and even desired motion planning cycle rates. The models can also be dynamic and adjust as the rover progresses through its mission, in response to changing terrain conditions or even partial failures of the locomotion system as outlined below.

3. Global Motion Planning in State Lattices

In our global motion planning field experiments, we validated our approach to global motion planning, previously described in [3][4]. This approach, briefly outlined in this section, consists of two components: the search space (along with methods for generating it) and the search algorithm itself. Most of the innovation resides in the former, while the latter can be any standard search algorithm.

3.1 Search Space Generation

Search spaces in global search typically consist of a simple primitive set that satisfies some or no mobility constraints. A grid is the simplest search space typically applied. By allowing instantaneous changes of direction, it does not satisfy smoothness (dynamic) constraints (Figure 4, top). A more sophisticated approach is the use of the controls of the Dubins or Reeds-Shepp car, where the vehicle mobility includes minimum turning radius turns (Figure 4, middle).

Our method is an improvement over this standard approach because it extends the set of motion primitives to include actions that are more sophisticated and expressive than those of the Dubins and Reeds-Shepp cars, while also enforcing consistency with a differential equation to guarantee smoothness of velocity and curvature (Figure 4, bottom). Another differentiator includes the non-uniform discretization of state, which relaxes the uniformity requirement, making it possible to choose convenient discrete values of state [3].

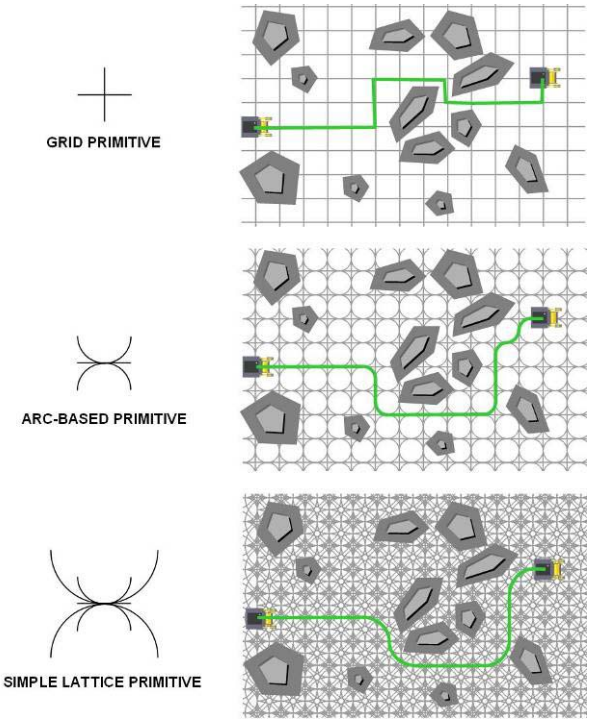

Figure 4. Search space comparisons. Three search spaces and their solutions navigating an obstacle field are shown. The grid-based solution generates a short path but incorporates abrupt heading changes. The arc-based primitive generates a solution that has smooth heading but discontinuous curvature. A simple lattice structure with curvature states and edges based on a dynamic trajectory generator is automatically feasibility everywhere.

3.2 Search Algorithms

The other component of our constrained motion planning method is a search algorithm. It finds the path, represented as a sequence of motion primitives, that leads the rover to the goal, while minimizing the desired measure of cost (time or distance of travel, energy expenditure, etc.). In general, any search algorithm can be applied to search the state lattice for a solution. In order to support efficient replanning, we utilize the D* Lite search algorithm. We apply the algorithm without any modifications, since the state lattice is a directed graph, similar to other applications of this algorithm. Our planner derives its advantages from the greater expressiveness and inherent feasibility of the state lattice.

One important implementation detail is the method for computing the cost of motions in the state lattice efficiently. In order to estimate this cost, it is necessary to simulate the behavior of the vehicle, subjected to the corresponding control in the environment. In mobile robotics applications, the cost of traversing the terrain is often represented via a 2D cost map. Then, the cost estimation can be achieved by summing the costs of the map cells covered by the robot's swath as it moves. Since the search space consists of known motion primitives, their swaths can be pre-computed. Hence, instead of the costly vehicle simulation, edge cost computation can be reduced to accumulating the cost over a pre-computed set of map cells. Figure 5 illustrates a simulated rover that navigates in a previously unknown environment using the presented approach. As the rover travels, it re-plans its path (blue) according to new obstacle information that comes within the extent of its perception (green circle).

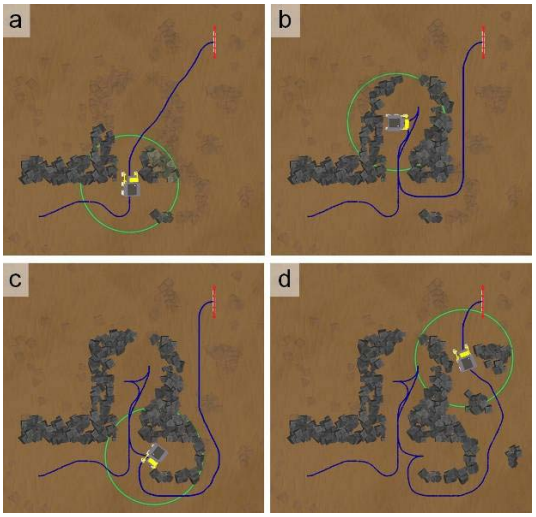

Figure 5. Simulation experiment in efficient replanning with a limited perceptual horizon using D* Lite. Note that the (always feasible) path adapts to the newly discovered knowledge about the environment. A turn-in-place and a multi-point maneuver were generated by the planner automatically.

4. Field Experiments

In this section, we follow the dual exposition above, where we first describe the results of our approach to the trajectory generation and follow with the results of the state lattice motion planner.

4.1. Model-Based Trajectory Generation Field Experiments and Results

The set of field experiments concerning the model-based trajectory generator were conducted with Rocky 8, the JPL planetary rover prototype previously appearing in Figure 1. Two significant results are presented in this section, including impaired mobility compensation and motion planning considering the all-wheel steering maneuverability in Sections 4.1.1 and 4.1.2 respectively.

4.1.1 Impaired Mobility Compensation

As previously mentioned, the ability to compensate predictively for model dynamics is important for robust motion planning in complex environments. We must understand the feasibility and costs associated with an action before we choose to execute it. This becomes more important when the mobility system

operates in a suboptimal or unexpected manner. We sought to demonstrate that the effects of an impaired drive wheel, as seen in the MER mission, could be compensated for predictively using the trajectory generator's motion model. Figure 6 illustrates the effect of a rover attempting to place itself in front of a target of scientific interest. In this situation, the back left wheel has been disabled, causing the vehicle to drag the wheel and pull the vehicle to the left during its execution of the motion. If a model of these effects were included in the motion model of the trajectory generator, the trajectory optimization would generate an action that would drive the vehicle harder to the right to compensate for the dragging of the impaired wheel.

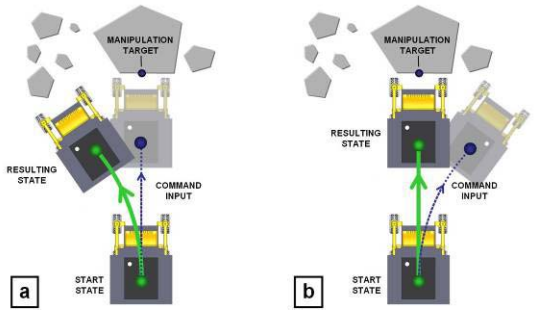

Figure 6. Model-based trajectory generation with impaired mobility models. A simple instrument placement task with a broken drive wheel (back left) is shown. (a) shows the result of not compensating for these actions whereas (b) shows the result of allowing the trajectory generator to optimize the input to meet the terminal boundary state constraints with knowledge of the impaired mobility system.

We were able to demonstrate this effect on Rocky 8 by manually disabling the back right wheel's drive motor on the platform for a simple instrument placement task (Figure 7). With the drive motor disabled, the rover drags this wheel, digging a trench and causing the vehicle to pull to the right. The result of executing the original action with the impaired mobility system is the vehicle positioned behind and to the right of the target terminal state. Just as in Figure 6, by incorporating a model of the effects of the impaired wheel, we can automatically generate a corrective action that drives the vehicle harder to the left to reach the target terminal state. This experiment did not use a path following controller (feedback) in order to demonstrate the capacity of feed forward to eliminate this model error disturbance before it occurs. A complete implementation would also use feedback to remove any remaining uncompensated error.

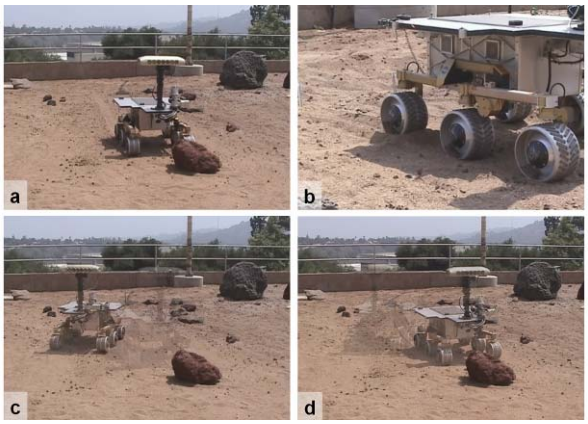

Figure 7. Field experiments of impaired mobility recovery. A simple instrument placement task (a) is attempted with the right rear drive motor disabled (b). The dragging wheel causes the vehicle to pull to the right (c). By including a model of these effects, a corrective action that drives the vehicle harder to the left (as in Figure 6) is generated and executed (d).

This result demonstrates the value of the flexibility of the trajectory generator's motion model and it is a good example of how this algorithm can enable more robust robot autonomy. For the application of planetary exploration, it is difficult if not impossible to repair damaged mobility systems. It will become necessary for mobile robots to understand the potentially changing capabilities of its mobility systems for missions with greater autonomy.

4.1.2 All-Wheel Steering Motion Planning

The ability to generate trajectories that can exploit the all-wheel steering maneuverability of the platform can lead to efficient, safe navigation in complex environments and to multiple target instrument placement maneuvers. For example, consider the situation in Figure 8 where the rover must navigate to a target in an obstacle field. If the robot can exploit the all-wheel steering maneuverability, it can generate more optimal (shorter, fewer reversals) trajectories [2]. The corresponding field experiment in Figure 8 can be found in Figure 9.

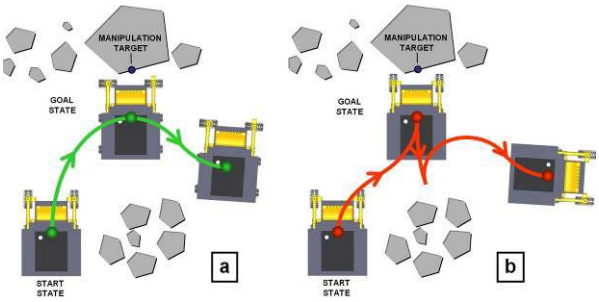

Figure 8. Single target instrument placement in cluttered environments. The all-wheel steering maneuverability can be exploited to generate safer and lower-cost trajectories. The all-wheel steering maneuver (a) reaches its target state by driving to and leaving with a direction 90° from the forward axis of the vehicle. The corner-steering maneuver comes much closer to the obstacles and must back up to leave the target.

Figure 9. Exploitation of the all-wheel steering maneuverability for efficient path generation.

Another important example is that of the multiple target instrument placement problem. If a scientist is interested in acquiring several samples in a small and confined space, it is appropriate to optimize the entire path that includes all boundary state pairs (Figure 10). In this example, it is much more effective to rely simply on the all-wheel steering maneuverability than to execute a longer action with forward and backward sections. Figure 11 exhibits the ability to generate trajectories that can exploit the all-wheel steering maneuverability of the platform on actual hardware.

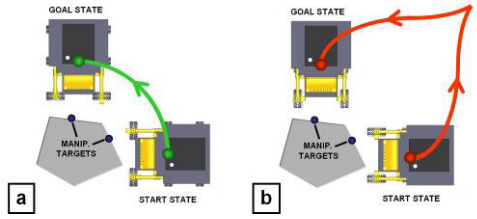

Figure 10. Multiple target instrument placement with all-wheel steering maneuverability. The two motion plans above demonstrate the efficiency gain by exploiting the all-wheel steering maneuverability (a) and not (b).

Figure 11. Exploitation of the all-wheel steering maneuverability for efficient path generation.

While the all-wheel steering maneuverability does not apply to the MER-type robots (they are limited to corner-steering mobility), other research platforms, including Rocky 8, FIDO, and the Athlete rover prototypes, exhibit this capability.

4.2. State Lattice Motion Planner Field Experiments and Results

In our experiments, the motion planner was capable of guiding the vehicle to a specified goal, while repeatedly and frequently re-planning, thanks to D*, to take into account the perception information acquired during execution of the plan. When coupled with several related components, the planner performed in the capacity of a complete rover navigation system. Additional components included:

- stereo perception for obtaining a 3D map of the environment,

- a terrain analyzer for computing traversability cost of the terrain, derived from Morphin [8],
- rover motion control, state estimation, telemetry and visualization provided by CLARAty.

The resulting system performed well and was capable of running all of the above modules plus our planner on-board, autonomously, and *continuously*, i.e. while interleaving planning and execution so that the rover did not stop moving for re-planning. This suggests that the proposed planner offers runtime that is acceptable for rover hardware and compares well with state of the art solutions.

Several implementation specific details are described below, followed by field results. The motion planner was implemented on the FIDO rover at the JPL Mars Yard. FIDO was equipped with a 1.6GHz Pentium M CPU and 512MB RAM on a PC104 stack, running VxWorks. Similar to Rocky 8, the rover features six independently steerable wheels. A pair of navigation cameras in the rover's mast was used for stereo perception, utilized for terrain analysis. The rover's motion control was performed in software in a centralized fashion by the main CPU (which certainly affected processor availability).

Several parameters that were used by the planner are specified in the Table 1. *Branching factor* refers to the number of edges that emanate from the lattice nodes. The particular state lattice used included four dimensions: translational coordinates (x and y), heading and curvature. The set of motion primitives that resulted from these choices of discretization is shown in Figure 12.

Table 1. Parameters utilized by the state lattice motion planner.

Branching Factor	45 – 60 (heading dependent)
Translational (x, y) Discretization	20cm cell size, uniform (square cells)
Heading Discretization	16 values, non-uniform: {0, atan2(1, 2), 45°, etc}
Min. Turning Radius	0.5m (FIDO platform)
Curvature Discretization	9 values, uniform

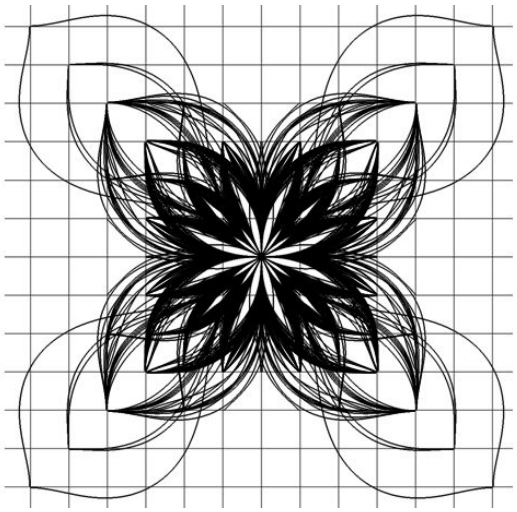

Figure 12. State Lattice Control Set Designed for Rocky 8 / FIDO Mobile Robot Platform.

Figure 13 illustrates re-planning as performed by the motion planner during navigation. Figure 13-A is a view of the planner cost map and a solution path early in the traversal, before some high-cost areas became visible to the perception system. Note that the planner was configured to operate in optimistic mode, following typical D* practice, where unknown areas were assumed to be traversable until determined otherwise. Figure 13-B is a view of the same experiment later in the traversal, where the path was visibly modified by the planner due to the newly discovered obstacles. Average runtime of the planner in the experiment featured above was 1.3 seconds. An illustration of the FIDO rover during our field experiments with the global planner is shown in Figure 14. The dotted line in the figure is the feasible, continuous-curvature path that the planner computed and the rover executed.

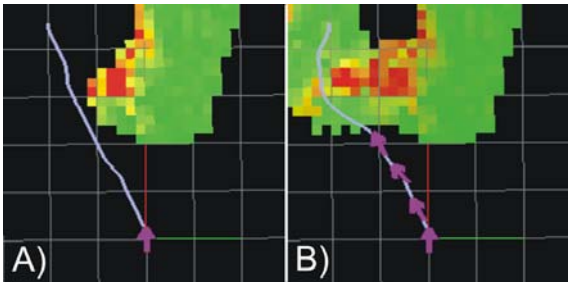

Figure 13. Sample of field experiments with the state lattice motion planner. A) shows an initial plan, and B) demonstrates re-planning due to new perception information.

Figure 14. FIDO rover during field experiments with the lattice planner.

In general, during multiple field trials, the planner exhibited re-planning time less than 1 second when there were relatively few perception updates, and several seconds when the updates were substantial. The planner runtime on the order of a second, exhibited on a rover CPU, which was shared with all other navigation components, suggests that the proposed method is competitive with prevailing approaches with respect to computational requirements.

5. Conclusion and Future Work

Autonomous rover navigation is an important technology for future missions of planetary exploration. In this paper, we have described the application of algorithms developed for motion planning in difficult or constrained environments, as applied to the problem of rover navigation. These methods offer unique capabilities, including the ability to invert general dynamics models and to exploit the full maneuverability of the motion platform, while exhibiting real-time or near real-time performance.

We intend to continue developing the algorithms described and referenced by this paper and apply them to the problem of rover navigation for planetary exploration.

6. Acknowledgments

This research was conducted at the Robotics Institute of Carnegie Mellon University under contract to NASA/JPL as part of the Mars Technology Program. We thank our colleagues Dr. Antonio Diaz-Calderon, Dr. Hari Nayar, and others in the JPL Robotics section and CLARAty team for their support in integrating, and testing our algorithms on the prototype research rovers.

7. References

[1] S.M. LaValle. *Planning Algorithms*, Cambridge University Press, 2006.

[2] T.M. Howard and A. Kelly. "Optimal rough terrain trajectory generation for wheeled mobile robots", Int. Journal of Robotics Research: 26-2, pp. 141-166, 2007.

[3] M. Pivtoraiko, R.A. Knepper, and A. Kelly.,"Optimal, smooth, nonholonomic mobile robot motion planning in state lattices", Technical Report CMU-RI-TR-07-15, Robotics Institute, Carnegie Mellon University, May 2007.

[4] M. Pivtoraiko and A. Kelly. "Multi-resolution motion planning using state lattices", Technical Report CMU-RI-TR-07-41, Robotics Institute, Carnegie Mellon University, November 2007.

[5] S. Goldberg, M. W. Maimone and L. Matthies, "Stereo vision and rover navigation software for planetary exploration", in Proc. of the IEEE Aerospace Conf., vol. 5, pp. 2025-2036, 2002.

[6] J. Carsten, A. Rankin, D. Ferguson and A. Stentz, "Global path planning on board the Mars Exploration Rovers", in Proc. of the IEEE Aerospace Conf., pp. 1-11, 2007.

[7] Kelly A et al., "Toward reliable off-road autonomous vehicles operating in challenging environments", *Int. Journal of Robotics Research*, 25: 5-6, pp. 449-483, 2006.

[8] R. Simmons, L. Henriksen, L. Chrisman and G. Whelan, "Obstacle avoidance and safeguarding for a lunar rover", in Proc. Of the AIAA Forum on Advanced Developments in Space Robotics, Madison, WI, 1996.

[9] A. Lacaze, Y. Moscovitz, N. DeClaris, and K. Murphy, "Path planning for autonomous vehicles driving over rough terrain", in *Proc. of the IEEE Int. Symp. On Intelligent Control*, 1998.