Fifth International Symposium on Artificial Intelligence and Automation in Space, Noordwijk, The Netherlands, 1-3 June 1999

ROAMS : Rover Analysis, Modeling and Simulation Software

Jeng Yen, Abhinandan Jain and J. (Bob) Balaram
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive, Pasadena, CA 91109

Abstract

In this paper, we present the development of **ROAMS** software for **real-time simulation** of mobile robotic vehicles. The purpose for the simulation is to provide a virtual testing ground for various subsystems and components of the robotic vehicle, which include a mechanical subsystem, an electrical subsystem, internal and external sensors, on-board control software. Using the **DARTS/DSHELL** framework, the real-time simulator can be applied to both operator-in-the-loop and off-line simulation. Its flexibility can accommodate the simulator to different levels of sophistication for various tasks in system engineering, software engineering, and for research scientists, maintenance and operation technicians. However, to achieve "real-time" in simulation of complex physical systems is a non-trivial task. Efforts have been made to build the **Rocky 7** rover model for efficient and stable simulation. Currently, the rover model is comprised of its mechanical, electrical, and sensor subsystems, and connected with the on-board software. With additional terrain models, we developed solution techniques that permit real-time simulation of the rover traversing **Mars-like terrain** on workstations. Further integration of ROAMS with a planning subsystem is also discussed in this paper.

kinematics

1 Introduction

ROAMS is based upon JPL's DARTS/DSHELL [4], a multi-mission spacecraft simulation software, for simulation of robotic vehicles. Inherent to DARTS/DSHELL is the development environment for modeling sensors, actuators, electrical and mechanical subsystems. Expanding these capabilities, ROAMS allows modeling of a closed-loop mechanism and contacts between vehicle's wheels and terrain. Efforts have been made to achieve real-time simulation of the rover traversing a class of Mars-like terrain. For the development purposes, we used a Mars rover prototype Rocky 7 [6] for a base model of the robotic vehicle.

The Rocky 7 Research Platform

The Rocky 7 rover configuration is shown in Figure 1. Like Sojourner, the wheel diameter of Rocky-7 is 13 cm. The mobility system is a modification of the Rocker-Bogey design used in previous rovers at JPL . It consists of two rockers hinged to the sides of the main body. Each rocker has a steerable wheel at one end and a smaller bogey at the other end. Unlike its predecessors Rocky-3 and Rocky-4 (and the Sojourner flight rover) that have four steerable wheels, Rocky-7 has only two. Rocky 7 has a closed-loop mechanism (rocker-differential-bogey) designed to give it high mobility

in rough terrain. The kinematics constraints for the closed-loop mechanism are given by fixing the distance of the loci of clevis attachment points. Measuring the clevis length, and solving for the rocker angle, we obtained an explicit form of the equations. The kinematic model is presented in 2, where the driving functions and internal constraints are described.

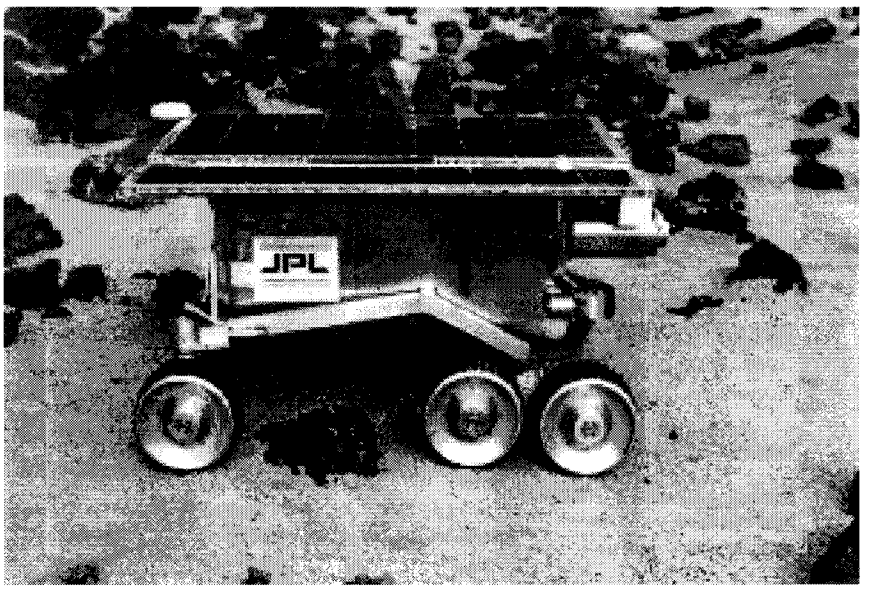

Figure 1: Rocky-7 Side View

Novel Solution Technique

One of the essential component of the rover simulation is the terrain, where the wheels of rover determine its configuration. We developed a novel numerical solution in ROAMS to treat the inverse kinematics that arise from driving rovers on the Mars-like terrain. The inverse kinematics of the rover is posted as a constrained optimization, where the objective function is composed of the weighted norm of the path-follower and kinematic loop-closure equations, and the constraints are the contact equations of the rover's wheels and terrain.

Additional constraints are a result of contacts between the wheels and the terrain. Depending on the contact rule, e.g., non-penetrative or penetrative, and the terrain profile, e.g., smooth or non-smooth, the constraint equations can be problematic for a Mars-like environment. For instance, all six wheels may not be in contact with the ground at all times, because of physical limitations, such as joint stops on the rocker-differential-bogey mechanism. This problem is resolved by using a bound-constraint method, which fixes the joint angles at their limits. In particular, the active contact condition is maintained if the Newton iteration is smoothly convergent. If the Newton iteration diverged, we check the joint limits and estimate the condition number of the iteration matrix to decide (1) acceptance or rejection of the current solution, (2) a new set of active contact constraints. This approach is very effective for solving the inverse kinematics in the preliminary test examples.

The terrain is represented by a parametric surface, and the rock profile is hemispherical. For both the smooth, (e.g. its profile consisting of functions with continuous second order derivatives) and non-smooth terrain, the above numerical process is applied with some modifications. On a smooth terrain, the solution is carried out using a Newton-type method, which is globalized by

applying the backtracking line search strategy. For non-smooth terrain, we apply an inexact Newton iteration with approximated iteration matrix and the same solution method as the smooth terrain. Using a decision-making process at the each iterative solution, the class of Newton-type methods can produce robust results for the seemingly ill-conditioned problem. The efficient solution yields a real-time application of rover simulation on a workstation computer platform. Detailed solution technique is contained in Section 3.

Applications of ROAMS

Building on this novel solution technique, we have applied rover simulator to testing the on-board software for Rocky 7. A navigation state machine was used driving Rocky 7 rover model against a terrain with randomly distributed rocks. Applying DARTS/DSHELL methodology, we implemented models for hardware devices, such as panoramic spectrometer, sun sensor, tilt sensor, obstacle detection camera, solar panel, battery, etc., producing feedback to the on-board subsystems. Based on the numerical solution of inverse kinematics, the hardware instrument models provide high-fidelity synthetic data to test the control and navigation code. This environment permits fast and better design, implementation, and test for the software development. Issues and solutions related to the integration of this real-time software in the testbed are discussed in Section 4.

In a recent project, ROAMS has also been integrated with a planning system for coordinating multiple rovers in performing complex tasks. Three rovers are managed by the planner for task modeling and refinement. The extension of ROAMS is to integrate the high-fidelity rover models with the Rocky 7 software and the planning system. For this purpose, we developed additional hardware models, including a collision avoidance model, a obstacle detection model, models in the power unit, and the capability of running multiple rovers in ROAMS. Owing to the stability and accuracy of the numerical solution, these device models can provide high quality sample data for the planning model. The integrated system exhibits great potentials for advanced applications in areas of design, engineering and planning of mobile robotic systems. The development of this module is presented in Section 4.

2 Rover Model Develpment

Coordinate Frames and Variables

Coordinate frames and variables are as defined in Figure 2. The unconstrained rover's degrees-of-freedom (dof's) are seen to be three translational, three rotational, three internal $(\gamma_0, \gamma_1, \gamma_2)$, two steering (λ_1, λ_2), and six drive $(\psi_1, \ldots, \psi_6)$. Contact interactions at each wheel constrain these dof's to result in the rover typically having two translational dof's (x,y) and one angular dof (heading) when in full contact with the ground.

Notice the closed kinematics chain consisting of the main differential, the spherical pinned joints on the near side of rover, the main rocker axis, the spherical pinned joints on the far side of the rover, and back to the differential. In order to derive the forward kinematics it is necessary to solve for the rocker axis angle in terms of the differential angle. An analytic solution is possible and is graphed in Figure 3, where the nominal position of the rocker axis angle on each side of the rover is taken to be zero. Notice that at large angles for the differential, the solutions are no longer symmetric. However, for small differential angles a symmetric linear approximation is possible and used henceforth.

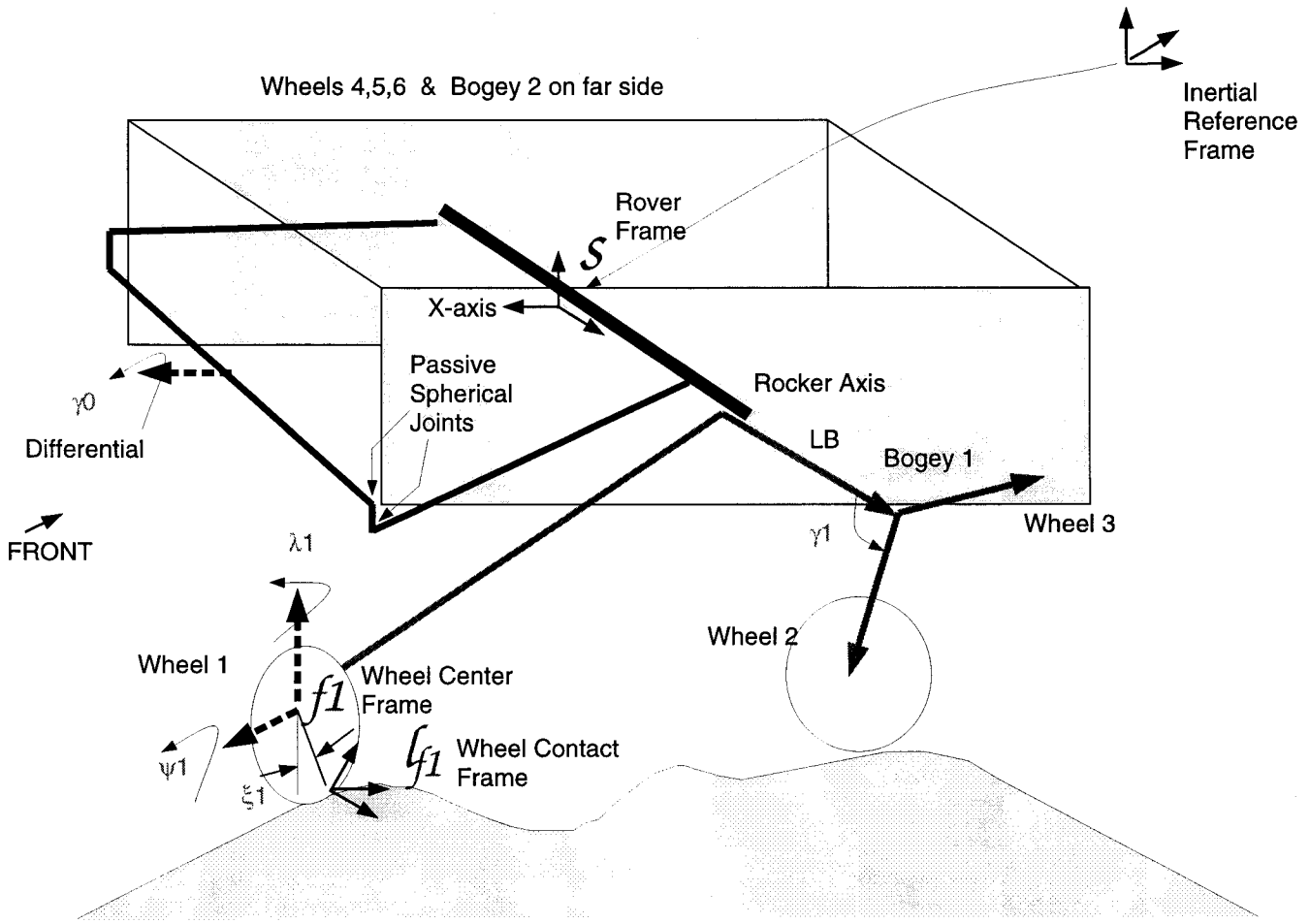

Figure 2: Rocky-7 Kinematics

Sensors, Actuators, Power Units and Scientific Instruments

On-board Rocky 7, sensors, actuators, cameras, and electrical subsystems are also modeled with the standard DSHELL library. The DSHELL models, navigation sensors, and custom electronics constitute a testbed for Rocky 7's system software.

The sun sensor, accelerometer, and wheel encoders are used to provide the internal odometery of the rover. The sun sensor and an obstacle detection CCD camera provide information for the autonomous path generation. The operator commands from the on-board software drive six wheel-motors and two steering motors. Extracting from the rover's position, ROAMS feeds back the odometer to its wheel encoders. By completing the loop of Rocky 7's autonomous path generation, ROAMS furnishes an effective testbed for the on-board navigation algorithm and software.

Rocky 7 has camera pairs with a 5 *cm* baseline at both ends of the vehicle, enabling bi-directional driving with obstacle avoidance. The camera model is implemented in ROAMS using a search of the synthetic rock field via bounding-box methods. Based on the height of the rocks in the approximated camera range, the signal of obstacle detection is sent to the rover navigation system. For multiple rovers, the obstacles include other rovers. When an obstacle is detected, the signal will trigger the vehicle navigator to generate an alternative path to the target location.

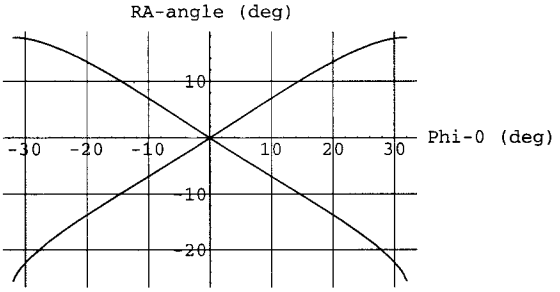

Figure 3: Solution of Internal Closed Kinematic Chain

The power unit of the rover consists of a battery model, a power draining model, and a solar panel model. Using the nominal power usage of the motors, electrical subsystems, an estimated power level of the battery is obtained. The solar panel model computes, using the panel's attitude and the sun position, accurate readings of the solar power generation. These data can be applied for the planning system, such as ASPEN (Automated Scheduling and Planning Environment). For testing and development of virtual rovers with this AI planner, a panoramic spectrometer was also developed for evaluating a geological scenario. The spectral measurements are generated using the position of the instrument and a pre-defined rock and mineral spectral model. More detailed descriptions of this integrated architecture are contained in [5].

Synthetic Terrain and Rocks Models

Two types of terrain, *smooth* and *piece-wise smooth*, are currently available in ROAMS. It is in the form of a parametric surface, where the elevation (z-coordinate) is a function of the x- and y-coordinates. On the terrain, a rock field can be applied to make up a synthetic Martian environment. The rock profile is represented by a half-sphere

$$z = \begin{cases} \sqrt{r - \sqrt{(x-x_c)^2 + (y-y_c)^2}}, & if \sqrt{(x-x_c)^2 + (y-y_c)^2} \leq r \\ 0, & otherwise. \end{cases} \tag{2.1}$$

where (x_c, y_c) is the center location, and r is the radius of the rock. When the contact location is within the rocks radius, equation (2.1) is the additional elevation of the wheel contact point. Note that the derivatives become increasingly degenerated as approaching to the rock's edge. Care must be taken when the wheel moving in and out the rock's circumference. This is dealt with a line-search strategy in the Newton iteration.

3 Computational Approach of Configuration Kinematics

It is straightforward to formulate the inverse kinematics of the rover traversing a terrain in the optimization problem:

$$\min_{q \in R^n} \quad f(q) \tag{3.1a}$$

$$subject\ to \quad g(q) \geq 0 \quad g \in R^k \tag{3.1b}$$

where $f(q)$ represents some *energy norm* of the kinematic constraints and driving functions, and $g(q)$ consists of the joint limits and wheel-terrain contact equations. For example, in the case of

path-following, the *energy norm* may be the weighted *L2* norm of the internal constraints and the path constraints, i.e., rover's body-coordinates (x, y) and heading angle $\boldsymbol{\theta}$ following the path. The constraints in equation (3.1b) are describing the distance function between terrain and the wheel and joint limits since we assume a non-penetrative contact of the wheel and the terrain.

Newton-Type Iterations

It is well-known that nonlinearity of the constraints (3.1b) can considerably increase the difficulties in the solution of (3.1) [7]. Furthermore, the objective of a real-time simulation requires an highly efficient solution. To achieve the required efficiency, we have applied the class of *first derivative methods* to (3.1), and solve the resultant nonlinear system via Newton-type methods. The gradient of (3.1a) and the constraints of (3.1b) yield the system of nonlinear equations of the form

$$F(q) = \begin{bmatrix} \nabla f(q) \\ g(q) \end{bmatrix} \tag{3.2}$$

where ∇f is the gradient of $f(q)$, and the components of $g(x)$ are of the form $w_i g_i(q)$ for some $i \in 1, 2, ..., k$ and $w_i \geq 0$. The variables $q \in R^n$ is made of the generalized coordinates of the rover model in the real-time computation engine DARTS [1]. In particular, $F \in R^m$ consists of the drivers, the internal constraints, and the contact equations. Based on the contact condition, e.g., non-penetrative or penetrative, and the terrain profile (smooth or non-smooth), $g(x)$ is to ensure that the rover is *always* in contact with the terrain. For each wheel, the contact is occurred with a positive *weight factor* w_i. During the iteration, w_i may be dropped to between 0 and 1, if a global search indicated the wheel may not in contact with the terrain. Another feature is the limitation of joints. This is implemented using the solution of a sub-problem, where the search direction of the corresponding joint variables (that reach the bound) is zero. These modifications furnish an effective numerical solution of (3.2) for the rover inverse kinematics, and permit the applications of real-time rover simulation.

Properties of Linear Search Algorithm

For a robust Newton convergence, we implemented a step selection strategy using backtracking, see [3], pp. 120-126. This algorithm ties a Newton step at the each iteration. For a smooth nonlinear system (3.2), the sequence of solution generated by the iteration will converge very fast, i.e., quadratically or superlinerly, to a local minimizer of equation (3.1). For the rover inverse kinematics, the application of backtracking line-search algorithm is particularly effective. As explained, the nonlinear system (3.2) is of a specific structure such that the nonlinear contact equations are loosely coupled with the mostly linear driving functions. This yields that the step selection should mimic the search of a contact point between the wheel and terrain. Thus, we enhanced the step selection using physical dimension of the wheels and metrics of the terrain. The resulting solution is so efficient that permits real-time rover simulation on a Sun Sparc 10 workstation.

Another feature associated with the step selection is the scaling of the search direction. With time, the Newton direction may become irregular when rover steps through a non-smooth region. For instance, at the edge of a rock on a flat terrain, or at the boundary of a piece-wise smooth patch, the Jacobian of equation (3.2) may become near ill-conditioned. The resultant search direction will therefore be irregular (often contains very large components). By scaling this vector into a reasonable size, the iterative solution may skip the local irregularity. In the preliminary test, it often reaches a near by solution that is good enough for the application.

Handling Terrain and Rocks

The terrain profile is a *parametric surface.* Each point on the terrain can be written as its Cartesian coordinate $[x, y, z(x, y)]^T$. On the ith road wheel, the *contact conditions* may yield

$$g_i^c = (x_w - x_c)^2 + (y_w - y_c)^2 + (z_w - z_c)^2 \tag{3.3a}$$

$$g_i^n = \tilde{t}_w t_c = t_w \times t_c \tag{3.3b}$$

where $[x_w, y_w, z_w]^T$ is the position on the wheel and the contact position on the terrain is $[x_c, y_c, z_c]^T$, and t_w, t_c are tangent vectors of the wheel and the terrain at the contact position, respectively. Equation (3.3a) is the contact condition, while equation (3.3b) is the *non-penetrative* condition that constrain two tangent planes at the contact points to be co-linear. For Mars-like terrain, equation (3.3b) is often eliminated because of non-smoothness in the terrain and rock field.

Equation (3.3a) is in fact an *inequality* because some wheel(s) may not be in contact with the terrain. Moreover, on a rough terrain, multiple contacts may occur that yields additional difficulties. These and related computational problems are dealt with using a nominal contact point and a modified line-search mentioned in the above. Choosing the contact point on the wheel to be $x_w = x_c$ and $y_w = y_c$, we simplified (3.3a) to the form:

$$z_c - z(x_w, y_w) - r_W = 0 \tag{3.4}$$

where $z(x_w, y_w)$ is the elevation of the contact location on the terrain, and r_W is the wheel radius. Using equation (3.4) for the contacts, the modified Newton iteration exhibited an enhanced convergence. For non-smooth terrain, we are developing an alternative search method to treat equation (3.4). Nevertheless, when applied to the implemented solution to piece-wise smooth terrain or flat terrain with additional rock field, we have efficiently obtained solutions.

As noted before, equation (2.1) induces a singularity to the Jacobian of (3.4 when the contact location is at its circumference. To treat this problem, we apply a regularization of the derivatives of equation (2.1)

$$\frac{dz}{dx} = \begin{cases} -\frac{(x-x_c)}{\varrho} & if\ \sqrt{(x-x_c)^2 + (y-y_c)^2} \le r - \epsilon \\ -\frac{(x-x_c)}{r\epsilon}, & otherwise. \end{cases} \tag{3.5a}$$

$$\frac{dz}{dy} = \begin{cases} -\frac{(y-y_c)}{\varrho}, & if\ \sqrt{(x-x_c)^2 + (y-y_c)^2} \le r - \epsilon \\ -\frac{(y-y_c)}{r\epsilon}, & otherwise. \end{cases} \tag{3.5b}$$

where $\varrho = \sqrt{(x-x_c)^2 + (y-y_c)^2}\sqrt{r - \sqrt{(x-x_c)^2 + (y-y_c)^2}}$ and ϵ is small.

4 Applications of ROAMS

Testing Tool for Rocky 7 On-board Software

One direct application is to use ROAMS for testing the Rocky 7 on-board software. For development purposes, the system software is ported from a real-time operating system to the Unix platform. The system clock is set to use the standard Unix system time, permitting the (simulated) sensor feedback being synchronous.

Upon the rover received command to move to a desired location, the navigation algorithm generates a sequence of *way-points*, turns the rover toward the goal, and executes obstacle avoidance

activities. Connected to ROAMS, the navigation state machine is tested against the synthetic terrain made of a flat base and the Viking Lander 1 rock field. The navigator of Rocky 7 produces the commands for its four wheel-motors, moving toward the next way-point [11]. ROAMS applies the commands to approximate the next position of the rover, then solves the inverse kinematics for the configuration of the rover. Using the position and attitude, the sensor outputs are obtained and sent back to the rover software system. A simple diagram illustrates the configuration of the testbed in Figure 4. The testbed results have given qualitative measurements to the robustness of the state machines by the software system.

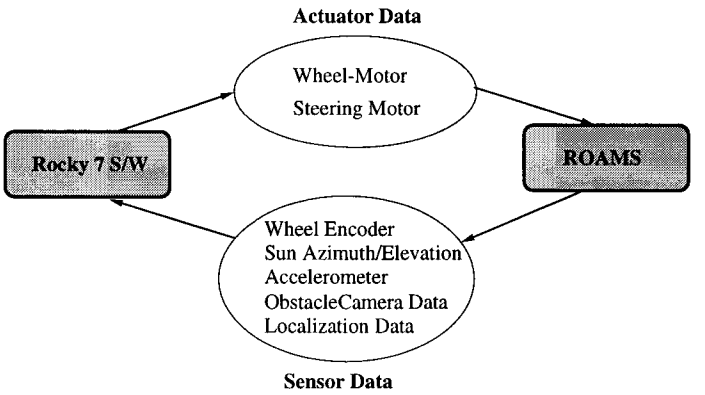

Figure 4: Rocky 7 Testbed Data Flow

Virtual Environment for Cooperative Rovers

ROAMS is also used for an integrated architecture being developed at the NASA Jet Propulsion Laboratory for utilizing multiple cooperating rovers. It provides a virtual testing ground for planning and coordinating multiple rovers in performing complex task for planetary surface exploration. For the multiple rover architecture, we extended the single rover simulation to support several rovers, and developed additional hardware models, which approximate the resources of each rover. In the current application of cooperating rovers, we evaluated the architecture using a model of terrain and a set of science goals over that model. Correspondingly, panoramic spectrometer model was developed to produce spectral measurements. Utilizing the development of Rocky 7 testbed, we constructed additional output data to the planning and scheduling system. As shown in Figure 5, the rover model contains the power unit, including a solar panel, a battery, and a power draining model. The solar panel model generates a voltage computed by the relative angle between sun position and the panel's surface. Power draining is based on the nominal power usage of electrical components. These and the battery model can give a high-fidelity prediction of the power level, allowing robust planning activities. It is shown through the development of this integrated architecture that ROAMS is a flexible and effective tool for modeling and testing of robotic vehicles.

Acknowledgments

The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

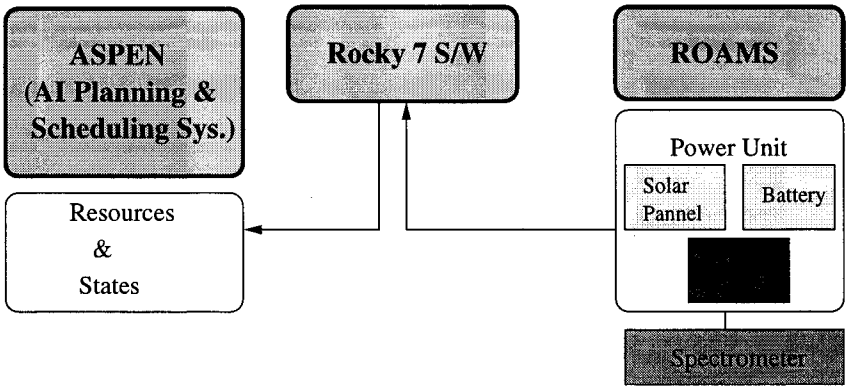

Figure 5: Cooperating Rovers Testbed Data Flow

References

[1] A. Jain and G. Man, *Real-time simulation of the Cassini spacecraft using DARTS: functional capabilities and the spatial algebra algorithm*, in 5th Annual Conference on Aerospace Computational Control, (Jet Propulsion Laboratory, Pasadena, CA.), Aug. 1992.

[2] R. S. DEMBO, S. C. EISENSTAT, AND T. STEIHAUG, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp.400-408.

[3] J. E. DENNIS AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1983.

[4] J. Biesiadecki, D. Henriquez and A. Jain, *A Reusable, Real-Time Spacecraft Dynamics Simulator*, in 6th Digital Avionics Systems Conference, (Irvine, CA), Oct 1997.

[5] T. Estlin, S. Hayati, A. Jain, J. Yen, R. Castano, R. Petras, D. Decoste, E. Tunstel, S. Chien, E. Mjolsness, and D. Mutz, *An integrated architecture for cooperating rovers*, to appear in proceedings of the symposium of iSAIRAS '99.

[6] S. Hayati, R. Volpe, et al., *The Rocky 7 Rover: A Mars Sciencecraft Prototype*, Proceedings of the IEEE International Conference on Robotics and Automation, Albuquerque NM, April 20-25 1997.

[7] P. Gill, W. Murray, and M. Wright, *Practical Optimization*, Academic Press Inc. (London) Ltd., 1981.

[8] D.J. Montana, *The kinematics of contact and grasp*, The International Journal of Robotics Research, Vol. 7, No. 3, June 1988.

[9] R.M. Murray, Z. Li and S.S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 1994.

[10] R. Volpe, J. Balaram, T. Ohm and R. Ivlev, *Rocky 7: a next generation Mars rover prototype*, Advanced Robotics, Vol. 11, No. 4, pp. 341-358 (1997).

[11] R. Volpe, *Navigation Results from Desert Field Tests of the Rocky 7 Mars Rover Prototype*, to appear in International Journal of Robotics Research, Special Issue on Field and Service Robots, 1999.