

Using Planning, Scheduling and Execution for Autonomous Mars Rover Operations

Tara Estlin, Daniel Gaines, Caroline Chouinard, Forest Fisher,

Rebecca Castano, Michele Judd, and Issa Nesnas

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
{firstname.lastname}@jpl.nasa.gov

Abstract

With each new rover mission to Mars, rovers are traveling significantly longer distances. This distance increase raises not only the opportunities for science data collection, but also amplifies the amount of environment and rover state uncertainty that must be handled in rover operations. This paper describes how planning, scheduling and execution techniques can be used onboard a rover to autonomously generate and execute rover activities and in particular to handle new science opportunities that have been identified dynamically. We also discuss some of the particular challenges we face in supporting autonomous rover decision-making. These include interaction with rover navigation and path-planning software and handling large amounts of uncertainty in state and resource estimations. Finally, we describe our experiences in testing this work using several Mars rover prototypes in a realistic environment.

Introduction

NASA has demonstrated that mobile robotic craft are a viable and extremely useful option for exploring the surface of other planets. The 2003 Mars Exploration Rovers (MER) have traveled across thousands of meters of terrain and gathered large amounts of valuable scientific data that is being used to answer many questions about the Martian environment. Future missions are being planned to send additional robotic explorers to Mars as well as to the moon and outer planets.

High-level decision making for these efforts, including for the MER Mission and the 1997 Mars Pathfinder mission, is performed on Earth through a predominantly manual, time-consuming process. For MER, a ground-based AI planning and scheduling tool is used to support science plan evaluation, however, a large team of

engineers is still required to perform a number of manual steps in order to generate a daily command sequence and uplink it to the rovers.

One significant problem with this approach to rover operations is that it can result in frequent underutilization of the robotic assets. Typically, command sequences are generated using conservative estimates on their time and resource requirements (e.g., overestimate time to drive to a new location to ensure that rover arrives by end of day). If the rover performs nominally or better than predicted, then the overestimates result in significant idle time for the rover, time in which the rover could be acquiring additional observations or sleeping to conserve energy.

If activities take longer to execute than expected then they will be aborted and future activities will be dropped. This can result in higher priority science observations not being performed because earlier observations ran long. Finally, new science opportunities can only be identified after scientists on the ground have been able to review downlinked data. This approach means many opportunities may not be realized, since once identified on Earth, the rover may have already traveled far past the object of interest. Further, not all image data can be downloaded thus some interesting terrain features may be completely missed. This case becomes even more prominent as rovers perform longer traverses (e.g., the MER rovers have driven more than 100 meters in a day).

A primary objective of our work is to use onboard planning, scheduling, and execution techniques to increase utilization of rover resources by enabling the rover to appropriately respond to unexpected problems and to take advantage of unanticipated opportunities. The Closed-Loop Execution and Recovery (CLEaR) system is intended to run with little communication with ground. It accepts science and engineering goals and creates a rover command sequence (or plan) that respects relevant constraints, while achieving as many goals as possible. The system executes the produced plan by dispatching commands to the rover's low-level control software and monitoring relevant state information to identify potential

problems or opportunities. If problems or new opportunities are detected, the system is designed to handle such situations by using re-planning techniques to add, move, or delete plan activities. Through this work, we have also identified a number of challenges for an onboard planning and execution system to not only produce valid plans, but also promote robust and efficient rover behavior. These challenges include properly interacting with the appropriate rover navigation software, handling uncertainty in state and resource estimations, as well as handling dynamic events, such as new science opportunities.

For the past several years, we have spent significant time testing the CLear system on several different rovers in the JPL Mars Yard. We will discuss our scenario designs for this testing and give an overview of the results including a discussion of how the system handled major scenario elements. Our main objectives for testing include simulating situations that might arise in future rover missions, (such as the Mars Science Laboratory or MSL mission, planned for launch in 2009), providing feedback on our approach, and identifying future directions that should be investigated. In the following section we outline some key challenges that we have identified for onboard decision-making software. Next, we present our current system approach and explain how this system fits into a larger rover architecture and other supporting software that contributed to our testing. We then describe several Mars rover scenarios, which were used to test our system on rover hardware, and describe how our system performed during that testing.

Challenges for Onboard Decision Making

Autonomous rovers have the potential for increasing science return by reducing rover idle time, reducing the need for entering *safe-mode*, and dynamically handling opportunistic science events without required communication to Earth. New missions are being designed that will require rovers to support more autonomous endeavors such as long-range traversals, complex science experiments, and longer mission duration. However, autonomy software designers face a number of challenges in providing software to support these types of operations. In this paper, we consider a few key challenges for using planning, scheduling and execution techniques to provide onboard decision-making capabilities.

To generate and/or modify its own command sequence for carrying out a set of science goals, the onboard planning and execution software will need to reason about a rich model of resource and temporal constraints. For example, it will need to predict power consumption of variable duration activities such as downlinks and traverses, keep track of available power levels, and ensure that generated plans do not exceed power limitations. When resources are over-taxed, the rover should be capable of making science/resource trade-offs in an effort to produce the highest science return. The rover will also

require execution and monitoring capabilities to carry out the generated plan on the rover platform. An execution system must be capable of commanding the control software, collecting state updates from sensors, monitoring plan behavior, and smoothly handling activity failures or unexpected events.

Over the course of a mission, the rover will be asked to perform a variety of science operations. The number and scope of these operations are typically limited by the rover onboard resources (e.g., power, memory, lifetime of hardware). Thus, science operations may have varying priorities that indicate their overall mission value. Onboard planning and execution software must reason about these priorities and handle newly identified science opportunities (which may be identified through onboard data analysis software) in a dynamic and efficient manner. For instance, the value of newly identified science observations must be weighed against current resource availability and other scheduled activities.

Sequence generation for rover surface missions also raises a number of interesting challenges regarding spatial reasoning capabilities. One of the dominating characteristics of rover operations is traverses to designated waypoints and science targets. This element is especially important in future missions that intend to explore large geographic areas. Onboard planning and execution software needs to coordinate with several levels of rover navigation software to generate an efficient and achievable rover plan. This coordination will likely include querying a path planner for route information, using position estimates to track rover progress, and correctly modifying the plan when navigation and obstacle avoidance software cause the rover to move off the predicted route.

Another predominant challenge in developing onboard autonomy software is dealing with the inherent uncertainty in predicting rover navigation and science operations. The difficulty is compounded by the tight resource and time constraints that a rover typically faces. At the resource and temporal level, the estimation of items such as power, memory and even activity duration can be highly uncertain. Rover missions are directed at exploring unknown planetary terrains. Requirements for traversing these new terrains are hard to predict. For instance, it is unknown what type of sand consistency a rover will be traversing, which can dramatically affect the required duration and power for a traverse. Similarly, the duration and resource requirements for science operations can vary as well. These variations could be simple, such as a lower than expected image compression ratio, or more complex, such as a drilling operation taking more power and time than originally estimated.

Furthermore, at the state level, since rovers lack an absolute positioning system, the uncertainty in the estimate of the rover pose grows with the distance traversed. This growing uncertainty creates a constant source of error in the knowledge of rover pose. The Sojourner rover used dead-reckoning and a single z-axis gyroscope to estimate

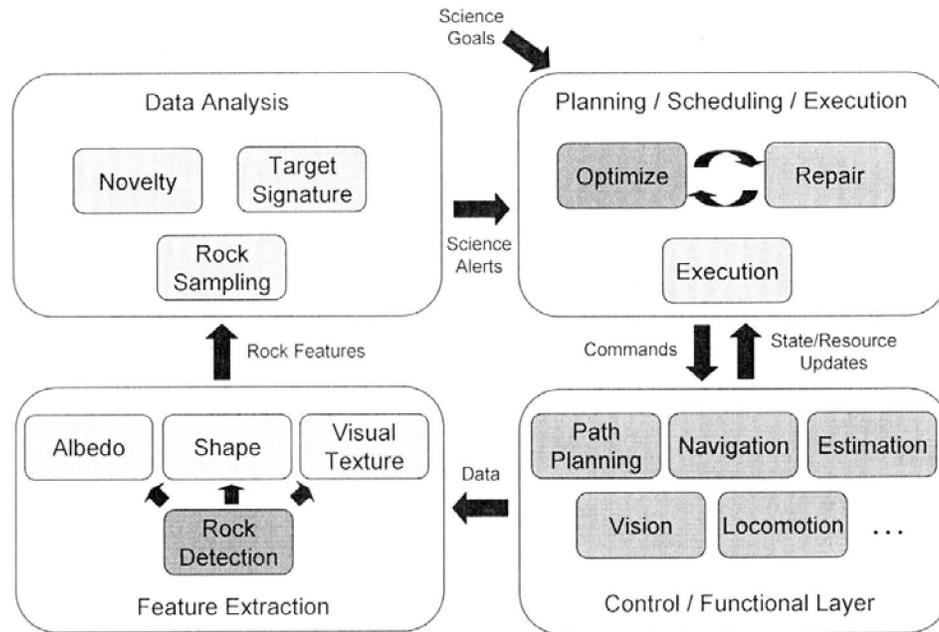


Figure 1: Onboard intelligent decision-making system framework. This framework shows how different decision-making capabilities interact. This paper focuses on the planning, scheduling and execution element of this framework.

rover position, which produced a position error of roughly 5-10% of distance traveled and an average heading drift of 13 degrees per day of traverse (Mishkin, et al., 1999). The MER rovers use more sophisticated techniques to provide position estimation, including a 3-axis gyro and visual odometry. However, these rovers still accrue significant position estimation error and on the ground localization software is often used to recalculate position. Since a large part of a rover schedule consists of rover moves to different locations, the onboard autonomy software must use estimations of position to predict the duration and resource requirements of different operations. If these predications are inaccurate, the autonomy software must be able to continuously modify the schedule to handle the uncertainty in the knowledge of actual rover position.

Planning, Scheduling, and Execution for Rover Operations

To address the issues outlined in the previous section, we have developed a system for high-level decision-making capabilities for future Mars rovers. The overall system framework and data flow is shown in Figure 1. This paper primarily focuses on the planning, scheduling and execution element of this framework, which provides autonomous rover command-sequencing capabilities. Other components will only be briefly described, but are further detailed in related publications.

CLEaR System

In this framework, planning, scheduling, and execution techniques are applied to provide rover-plan generation, execution, and monitoring, and the continuous modification of that plan based on changing operating context and goal information. These capabilities are provided by the CLEaR (Closed-Loop Execution and Recovery) system (Fisher, 2002; Estlin, 2005). CLEaR was developed to pursue a tight integration of planning and execution capabilities. To provide these capabilities, CLEaR closely integrates the CASPER (Continuous Activity Scheduling, Planning, Execution and Re-planning) continuous planner and the TDL (Task Description Language) executive system, which are described further below.

In our system framework, CLEaR handles the following functionality:

- Creating an initial plan based on an input set of goals
- Maintaining resource, temporal and other rover operability constraints
- Executing a plan by interacting with basic rover control functionality, such as navigation, pose estimation, locomotion and stereo vision
- Monitoring plan execution to ensure plan objectives are met
- Dynamically modifying the current plan based on plan activity, state and resource updates

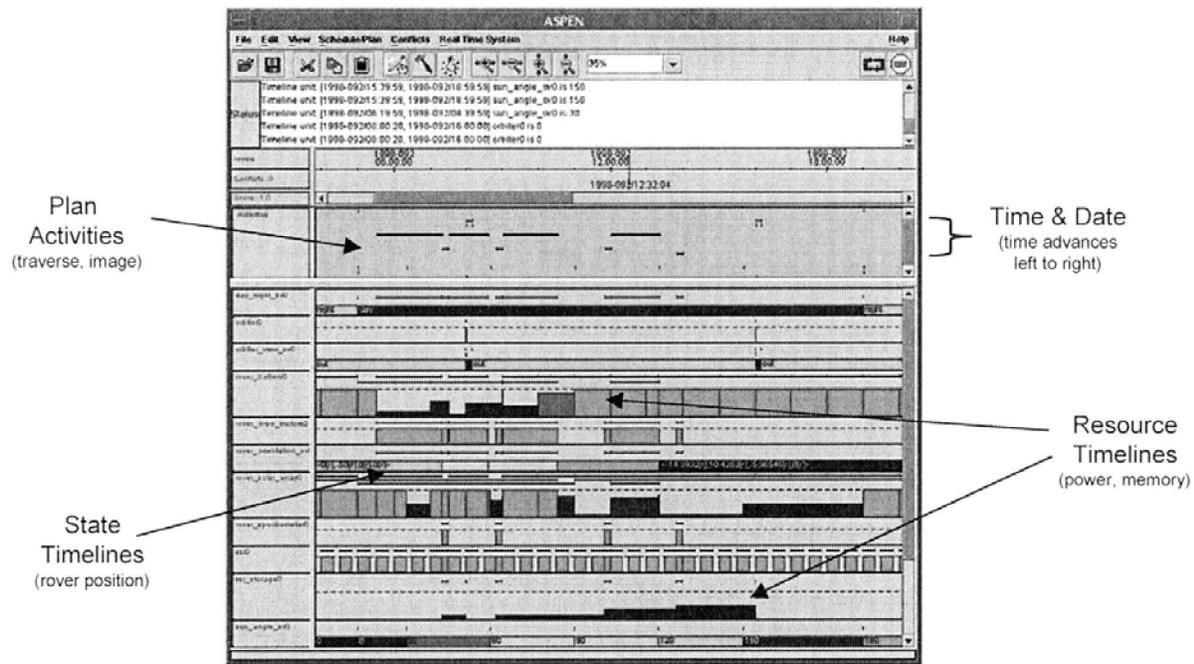


Figure 2: Sample rover plan displayed in planner GUI. Plan activities are shown in upper portion of window, where bars represent the start and end time of each activity. State and resource timelines are shown in bottom portion of the screen and show the effects of the plan as time progresses. Time is depicted as advancing from left to right.

- Performing plan optimization to reason about soft constraints and goal priorities
- Handling newly identified science goals (called *science alerts*) that are generated through onboard data analysis

Planning in CLEaR is provided by the CASPER continuous planning system (Chien, et al., 2000). Based on an input set of science goals and the rover's current state, CASPER generates a sequence of activities that satisfies the goals while obeying relevant resource, state and temporal constraints, as well as additional flight rules. Plans are produced using an iterative repair algorithm that classifies conflicts and resolves them individually by performing one or more plan modifications. CASPER also monitors current rover state and the execution status of plan activities. As this information is acquired, CASPER updates future-plan projections. This update may cause new conflicts and/or opportunities to arise, requiring the planner to re-plan in order to accommodate the unexpected events. An example of a rover plan displayed in the CASPER GUI is shown in Figure 2.

The executive functionality in CLEaR is performed by the TDL executive system (Simmons and Apfelbaum, 1998). TDL was designed to perform task-level control for a robotic system and to mediate between a planning system and low-level robot control software. It expands abstract tasks into lower-level commands, executes the commands, and monitors their execution. It also provides

direct support for exception handling and fine-grained synchronization of subtasks. In CLEaR, TDL also handles relaying appropriate activity and state data to CASPER, so that CASPER can adjust its plan accordingly. TDL is implemented as an extension of C++ that simplifies the development of robot control programs by including explicit syntactic support for task-level control capabilities. It uses a construct called a task tree to describe the tree structure that is produced when tasks are broken down into lower-level commands.

One of CLEaR's primary objectives is to provide a tightly coupled approach to coordinating goal-driven and event-driven behavior. Many past approaches have followed a three-level architecture style where the planning and executive processes are treated as *black box* systems. This is in contrast to how CLEaR enables the planner and executive to interact with each other and more effectively share the responsibility for decision making. In part this is managed through shared plan information and continual updates of state being made available to both the planner and executive. CLEaR also provides heuristic support for deciding when certain plan conflicts should be handled by the planner vs. the executive. For instance if a rover gets off track during a traverse, the reaction of the planner and executive need to be coordinated. If the executive believes it can resolve the navigation delay within the planned time constraints it will manage the plan changes. However, once the executive identifies that the repair will require more time or resources than allotted by

the planner, it will then fail the task, which will result in the planner using its global perspective to fix the problem.

Currently, CLEaR has a separate planner and executive and thus does share similarities to other three-layer architecture approaches. However, as compared to these approaches where planning is typically done in a batch fashion and takes on the order of minutes to hours, this integration uses a continuous planning approach, where plans are updated and repaired in a matter of seconds. This enables CLEaR to use planning techniques at a finer timescale for tracking the progress of plan execution, quickly identifying potential problems in future parts of the plan, and responding accordingly. As we expect minor portions of the plan to change frequently, we use a lightweight plan runner to dispatch activities to the executive a few seconds before the task's scheduled *start time*. This approach differs from the more common batch approach of turning the entire plan over to the executive for execution. Executive techniques are then used in reactive situations or at times where procedural reasoning is preferred (e.g., using a looping construct to represent the act of trying to grasp a rock, which may need to be repeated several times).

Another way that CLEaR differs from previous approaches is in how the delegation between the planner and the executive is managed. We have primarily taken a planning centric approach to this management. The planner handles the decision of when an activity should be sent to the executive as well as when to perform re-planning. Once the planner has mapped a planning activity to an executive task for execution, control over that one task is given to the executive. The executive may then perform further task expansions as a result of updates and/or exception handling. The executive also provides task completion status back to the planner by either marking an activity as completed or failed. A task is marked as completed when the executive decides the task has met its objective, or marked as failed if the executive concludes that relevant constraints cannot (or even might not) be met. The re-planning process is driven by applying and propagating updates to the plan, and then taking corrective actions to address any conflicts or opportunities that may arise. Re-planning can also be performed synchronously with any already executing task.

Problem Recovery

Due to the uncertainty of science and especially drive operations, a number of things can go wrong in the currently executing rover plan. Recovering from problems or other plan failures was the first general area to which we applied the CLEaR system. One type of plan problem this system addresses is over-subscription of resources (such as power, memory) or time. A number of elements can contribute to this problem, including terrain variability, rover hardware degradation and data compression inconsistencies. If plan activities end up taking additional resources or time during execution, the onboard system must ensure that critical activities that occur later in the

plan will still be able to correctly execute. To handle these situations, CLEaR continually tracks the state of the plan including activity completion times and resource usage. If problems (or conflicts) are introduced into the plan based on recent state data, CLEaR can move or delete activities to accommodate the changes. For example, if more power is used on a drive than expected, the system may need to delete several low priority science goals to ensure that enough power will be available for an end of day communication with Earth.

Another problem that CLEaR addresses is when obstacles in the terrain require plan changes to successfully visit goal targets. CLEaR currently employs a set of TSP heuristics to order science target visits in an optimal fashion. However terrain information may be incomplete when an initial plan is generated. During a rover mission, images from the rover navigation cameras can be used to build navigation and obstacles maps, however obstacles can be missed due to poor stereo or obstructions in the rover's sight path. If during plan execution, CLEaR determines that the drive to a particular target is running significantly behind schedule, it may re-evaluate the current target ordering to determine if a new, more optimal ordering can be found or if targets need to be deleted. This re-evaluation can also use any new terrain map information that may have been gathered from the rover's current position.

Science Alerts

To handle opportunistic science, we extended CLEaR to recognize and respond to *science alerts*, which are new science opportunities detected by onboard science-data analysis software. For example, if a rock is detected in navigation imagery that has a previously unseen texture or shape, a science alert may be generated to take additional measurements of that rock. Currently, science alerts can have different levels of reaction from the CLEaR system. The most basic reaction is to adjust the rover plan so that the rover holds at the current position and the flagged data is sent back to Earth for further analysis at the next communication opportunity. The next level of reaction is to collect additional data at the current site before transmitting back to Earth. Further steps include having the rover alter its path to get closer to objects of interest before taking additional measurements. These operations would provide new data that could not be obtained through analysis of the original image.

Plan Optimization

To reason about goal priorities and other soft constraints we used the CASPER optimization framework to continually search for a higher quality plan. User-defined preferences are used to compute plan quality based on how well the a plan satisfies these preferences. Optimization proceeds similarly to iterative repair. For each preference, an optimization heuristic generates modifications that could potentially improve the plan score. A modification is

then selected and applied to the plan. After a set number of iterations, the plan with the best score is selected to replace the current plan.

One key area where plan optimization was used was to take advantage of extra time or resources in the schedule. Since traverse times and rover resource usage are difficult to predict, it is often the case that a rover operation takes less time or power than expected. For instance, a traverse could take much less time than expected due to a benign terrain. For these cases, the optimization framework was used to dynamically add additional science goals to the plan that could not be fit in the original plan due to time and resource constraints. This capability enables the scenario where scientists on the ground specify a number of prioritized science goals, but not all of them may be achievable due to limited rover resources. However, some goals may be fit into the plan as time progresses due to resource usage being lower than predicted.

CLEaR also uses the optimization framework to decide how to respond to science alerts. Because it may not be possible to accommodate all alerts, a science alert is represented as an optional goal, which indicates its achievement is not mandatory but may improve the plan's optimization score if included in the plan. Before attempting to handle a science alert, CASPER protects the current plan by saving a copy before optimization. If the quality has not increased after a certain time limit, the previous plan is restored. If CASPER can handle a new science alert (e.g., by adding additional science measurements) without causing other negative affects, such as resource over-subscriptions or the deletion of ground-specified science goals, then the new plan that accommodates the science alert is used.

We created a set of plan modification functions that are invoked when the optimizer attempts to satisfy a science alert. How the plan is modified depends on the type of alert that is considered. When a science alert is received that requires holding at the current position until data is communicated with earth (called a *stop and call home* alert), the planner alters the plan to remove any non-engineering critical activities and wait for the next communication opportunity. If activities are currently executing, the planner requests the executive component of CLEaR to abort them. If activities are scheduled in the future, the planner deletes them and resolves any inconsistencies created by these deletions.

To handle a science alert that requests additional measurements (called a *data sample request* alert), the planner must generate a plan that achieves the new goals without deleting existing activities or causing conflicts that cannot be resolved (e.g., scheduling more activities than can be executed in a certain time window). To handle a data sample request, the planner must be able to add a new science observation and a new move command to correctly place the rover in position to take the observation.

Science Data Analysis

The Feature Extraction and Data Analysis modules, shown in Figure 1, are responsible for onboard science alert generation. Together with the planning and scheduling component, these capabilities comprise the OASIS onboard science system (Castano, et al., 2006). OASIS enables the rover to perform onboard analysis of collected science data and to trigger science alerts if interesting science opportunities are detected. For instance, if a rover is performing a long traverse, OASIS can analyze navigation images as they are taken to search for interesting rocks or other terrain features that the rover is passing.

As shown in Figure 1, new science data is first processed by the Feature Extraction component. Currently, we have focused on analyzing rocks (and other terrain data) within image data, but plan to expand to other types of data, such as spectrometer measurements. Images are broken down by first locating individual rocks, and second, by extracting a set of rock properties (or features) from each identified rock. Extracted rock properties (e.g., shape, albedo, visual texture) are then passed to the Data Analysis component of the system. This component consists of different prioritization algorithms, which analyze the data by searching for items such as rocks with features that match pre-known signatures of interest (identified by scientists on Earth), or novel rocks (i.e., outliers) that have not been seen in past traverses. If the analysis component detects new science opportunities of significant interest, it will generate a *science alert* that is sent to the planner.

CLARAty Robotic Architecture

The planning, scheduling, and execution component is also integrated with the Coupled Layered Architecture for Robotic Autonomy (CLARAty) (Nesnas, et al., 2006). CLARAty was developed at JPL to simplify the integration and testing of different robotic technology software on multiple hardware platforms and it provides a large range of basic robotic functionality. Through CLARAty, the CLEaR system has been tested with several JPL rover platforms, including Rocky 7, Rocky 8, and FIDO, which are shown in .

To run realistic scenarios with rover hardware, a number of supporting pieces of software were used. These components were provided through CLARAty and could run on the relevant JPL rover platforms. This software includes the Morphin navigation system (Urmson, et al., 2003), which enables the rover to avoid obstacles and navigate to specified waypoints, a position estimation algorithm, which integrates IMU (Inertial Measuring Unit) measurements with wheel odometry to estimate rover position and attitude (roll, pitch and heading), and other software that provide mobility and stereo processing.



Figure 3: Rocky 8 rover (left), FIDO rover (middle), Rocky 7 rover (right)

System Testing

To evaluate our system we have performed a large number of tests both in simulation and using rover hardware in the JPL Mars Yard. These tests covered a wide range of scenarios that included the handling of multiple, prioritized science targets, limited time and resources, opportunistic science events, resource usage uncertainty causing under or over-subscriptions of power and memory, large variations in traverse time, and unexpected obstacles blocking the rover's path.

Our testing scenarios typically consisted of a random number of science targets specified at certain locations. A map was used that would represent a sample mission-site location where data would be gathered using multiple instruments at a number of locations. Figure 4 shows a sample scenario that was run as part of these tests. This particular map is of the JPL Mars Yard. The pre-specified science targets (shown in Figure 4 as the larger circles) represented targets that would be communicated by scientists on Earth. These targets were typically prioritized and for most scenarios, constraints on time, power or memory would limit the number of science targets that could be handled. A large focus of these tests was to improve system robustness and flexibility in a realistic environment. Towards that goal we used a variety of target locations and consistently selected new science targets and/or new science target combinations that had not been previously tested.

A primary scenario element was dynamically identifying and handling opportunistic science events. For these tests, we concentrated on finding rocks with a particular target signature, which was described through specifying a target rock albedo level and shape. If rocks were identified in hazard camera imagery that had a certain interest score, then a science alert was created and sent to the planner. Science alerts would typically come in during rover traverses to new locations, but it was also possible for them to come in while the rover was at a science target location due to a small lag caused by image processing time. If a science alert was detected, the planner attempted to modify the plan so an additional image of the rock of interest would be acquired. A sample image that was taken in response to a science alert for a rock with low albedo (i.e., light colored) is shown in Figure 5.

Other important scenario elements included adding or deleting ground-specified science targets based on resource under or over-subscriptions. For instance, in some tests, the rover covered distances more quickly than expected and the planner was able to add in additional science targets that could not be fit into the original plan. Conversely, in other tests, the rover used more power than expected during traverses or science activities, which often caused a power over-subscription, where enough power was not being preserved for later plan activities. The planner resolved this situation by deleting some lower priority science targets. Unexpected energy drops during a traverse could also be handled by the executive, which detects the shortfall and stops the current traverse if there is not enough energy to complete it. In all cases, the planning and execution system attempts to preserve as many high priority science targets as possible while still adhering to required resource and state constraints.

Testing in Simulation

Since testing with rover hardware can be an expensive and time-intensive process, we ran a large number of tests in simulation using a relatively simple simulator. This simulator could execute rover sequence commands and simulate their effects at a coarse level of granularity. For instance the simulator handled items such as rover position changes and energy usage over straight-line movements, but did not simulate obstacle avoidance or rover kinematics. Another capability that was used in simulation was triggering multiple science alerts at pre-set or random times. This capability helped in evaluating the planner's capacity to correctly handle different opportunistic science scenarios.

To easily run and evaluate large numbers of tests, we also invested in a testing infrastructure, which allowed tests to be run offline and statistics automatically gathered, including information such as number of plan conflicts found and resolved, plan generation and re-planning time, number of goals satisfied, overall plan traverse distance and plan optimization scores. This testing infrastructure also enabled the automatic creation of mpeg movies that showed plan changes using snapshots of a plan visualization tool. This tool showed the results of plan generation and execution on an overhead map of the

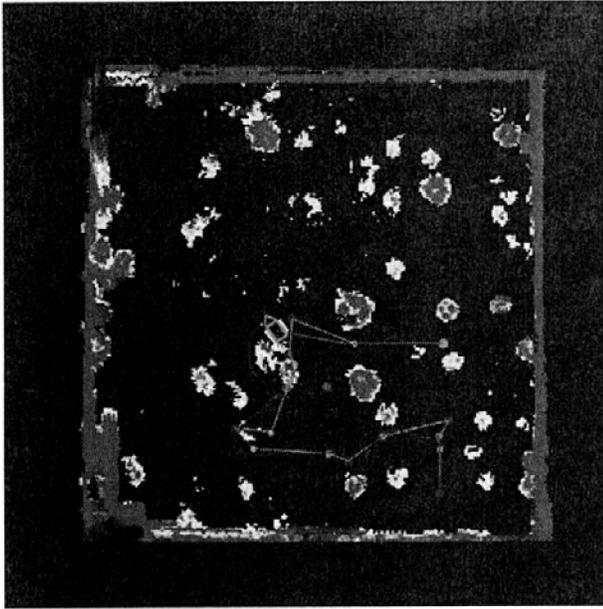


Figure 4: Sample plan shown in the Grid Visualization Tool (GriViT). Green lines show the planned path of the rover; blue lines shown the real path; and pink lines show the path that is currently executing.

world, and could be used for both simulated and hardware testing. An example plan snapshot displayed by this tool is shown in Figure 4. Planning and execution results were evaluated by examining gathered statistics and by viewing created mpegs to flag incorrect or non-optimal behavior.

Testing with Rover Hardware

In addition to testing in simulation, a large number of tests were run in the JPL Mars Yard (shown in Figure 6) using different rover hardware platforms. For the past several years the FIDO rover (shown in Figure 3) was used for the majority of tests. FIDO is an advanced technology prototype rover similar to the Mars Exploration Rover (MER). FIDO's mobility sub-system consists of a six-wheel rocker-bogie suspension capable of traversing over obstacles up to 30 cm in height. All demonstrated software has been designed to run onboard the rover, however during testing, only functional-level CLARATy modules, such as navigation and vision, and the OASIS rockfinding software were run onboard FIDO. Other modules, including the planning and execution module and the analysis module, were run on offboard workstations that communicated with the rovers using wireless ethernet, since a port of these components to the onboard operating system (VxWorks) was not complete.

Tests in the Mars Yard typically consisted of 20-50 meter runs over a 100 square meter area with many obstacles that cause deviations in the rover's path. Science measurements using rover hardware were always images,

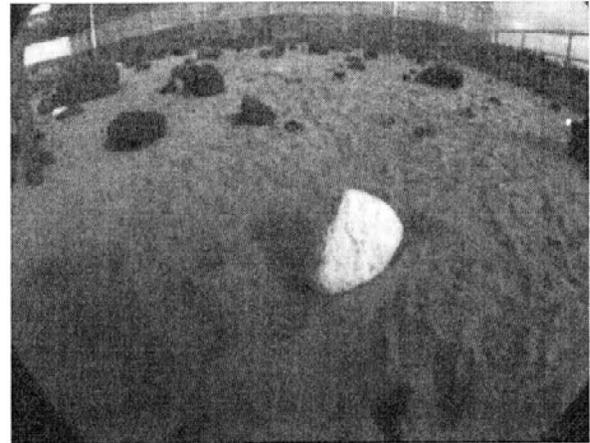


Figure 5: Sample image that was taken in response to a science alert for a rock of low albedo using the JPL FIDO rover.

since other instruments were not readily available (e.g., spectrometer). However different types of measurements were included when testing in simulation.

Testing in simulation and with real hardware provided important steps in the evaluation of our system. Many bugs were caught early through simulated testing, but others did not surface until significant runs had been performed on rover hardware. Furthermore, running with hardware often allowed a perspective that was difficult to attain through simulated testing. For example, the accuracy of rover turns towards new science opportunities was much easier to judge when running with hardware.

Related Work

A number of planning and executive systems have been successfully used for robotic applications and have similarities to the approach we describe in this paper. Most of these approaches have used some combination of planning and execution, however they differ in not only the behavior of these individual components, but also in how these systems interface with each other and with other system modules.

The Autonomous Sciencecraft Experiment (ASE) (Chien, et al., 2005) has demonstrated the capability of planning and data analysis systems to autonomously coordinate behavior of the EO-1 Earth orbiting satellite. ASE can also detect and respond to new science events, however it uses very different detection and analysis algorithms. The Remote Agent Experiment (RAX) (Jonsson, et al., 2000) was flown on the NASA Deep Space One (DS1) mission. It demonstrated the ability of an AI planning, execution and diagnosis system to respond to high-level spacecraft goals by generating and executing plans onboard the spacecraft. However, RAX did not incorporate data analysis to identify new science targets

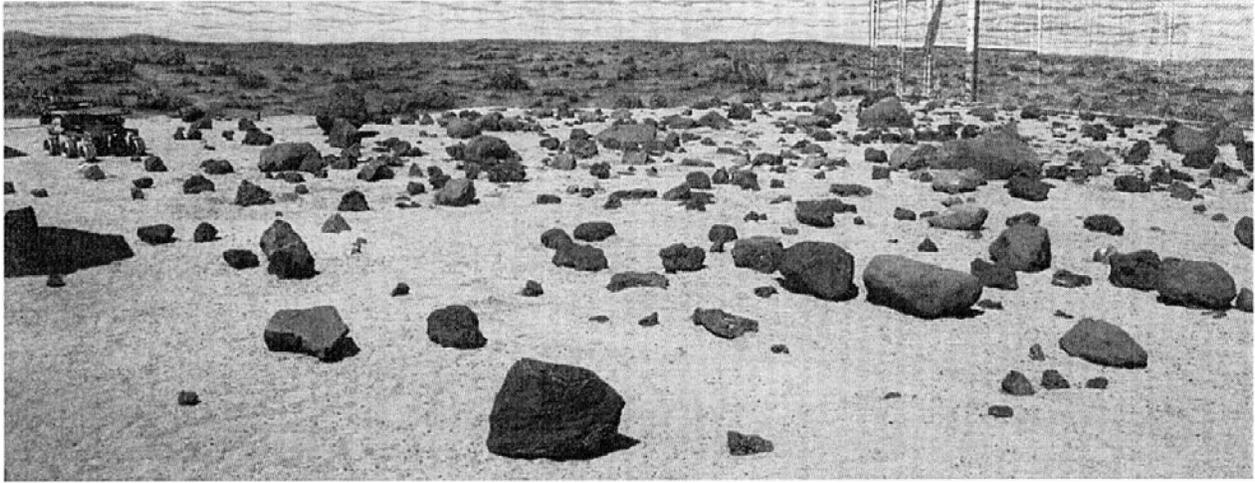


Figure 6: The JPL Mars Yard with terrain of various difficulties.

and used a batch approach to planning. Furthermore, since RAX and ASE were applied to spacecraft, neither handle issues associated with the uncertainty of surface navigation.

Another approach directed towards rover command generation uses a Contingent Planner/Scheduler (CPS) that was developed to schedule rover-scientific operations using a Contingent Rover Language (CRL) (Bresina, et al., 1999). CRL allows both temporal flexibility and contingency branches in rover command sequences. Contingent sequences are produced by the CPS planner and then are interpreted by an executive, which executes the final plan by choosing sequence branches based on current rover conditions. In this approach, only the executive is onboard the rover; planning is intended to be a ground-based operation. Since only a limited number of contingencies can be anticipated, our approach provides more onboard flexibility to new situations. In the CRL approach, if a situation occurs onboard for which there is not a pre-planned contingency, the rover must be halted to wait for communication with ground.

Other similar approaches include Atlantis (Gat 1991), 3T (Bonasso, et al., 1997), and a robotic control architecture developed at the LAAS-CNRS lab (Alami, et al., 1998) which all use a deliberative planner and an executive (or sequencing component) on top of a set of reactive controllers. These approaches have distinctly separate planning and execution techniques, have not closely interacted with navigation software used for rover missions, and are not integrated with onboard analysis system for dynamically identifying new goals.

Future Work

In future work, we plan to extend our capabilities for opportunistic science handling to include adding observations for different types of science instruments and

performing close-contact measurements for high priority alerts. We also will extend our system to handle the characterization of larger terrain features and areas that have been identified as important science targets. For instance, our system would handle science operations failing in different fashions such as an unsuccessful data acquisition (e.g., an over-exposed or miss-targeted image frame or an unsuccessful grasping of a rock).

Conclusions

This paper discussed a number of challenges for using planning, scheduling and execution techniques to provide autonomous rover capabilities for future NASA missions. We described our approach for using an onboard decision-making system and explained how it provides capabilities for sequence generation, execution, monitoring, re-planning, sequence optimization, and opportunistic science handling. Through a series of tests in simulation and on rover platforms, we have demonstrated our system's ability to robustly respond to unexpected problems and take advantage of unforeseen opportunities, thus achieving higher utilization of rover resources.

Acknowledgements

This work was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

References

(Alami, et al., 1998) R. Alami, R. Chautila, S. Fleury, M. Ghallab, and F. Ingrand, "An Architecture for Autonomy,"

International Journal of Robotics Research, 17(4) April, 1998.

(Bonasso, et al., 1997) R. Bonasso, R. Firby, E. Gat, D. Kortenkamp, D. Miller, and M. Slack, "Experiences with an Architecture for Intelligent, Reactive Agents," *Journal of Experimental and Theoretical Artificial Intelligence Research*, 9(1), 1997.

(Bresina, et al., 1999) J. Bresina, K. Golden, D. Smith, and R. Washington, "Increased Flexibility and Robustness of Mars Rovers," *Proceedings of the International Symposium, on AI, Robotics and Automation for Space*, Noordwijk, The Netherlands, June 1999.

(Castano, et al., 2006) R. Castano, T. Estlin, D. Gaines, A. Castano, C. Chouinard, B. Bornstein, R. C. Anderson, S. Chien, A. Fukunaga, and M. Judd, "Opportunistic Rover Science: Finding and Reacting to Rocks, Clouds and Dust Devils," *Proceedings of the 2006 IEEE Aerospace Conference*. Big Sky, Montana, March 2006.

(Chien, et al., 2000) S. Chien, R. Knight, A. Stechert, R. Sherwood, and G. Rabideau, "Using Iterative Repair to Improve the Responsiveness of Planning and Scheduling," *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling*, Breckenridge, CO, April 2000.

(Chien, et al., 2005) S. Chien, R. Sherwood, D. Tran, B. Cichy, G. Rabideau, R. Castano, A. Davies, D. Mandi, S. Frye, B. Trout, S. Shulman, D. Boyer, "Using Autonomy Flight Software To Improve Science Return on Earth Observing One," *Journal of Aerospace Computing, Information, and Communication*, April 2005.

(Estlin, et al., 2005) T. Estlin, D. Gaines, C. Chouinard, F. Fisher, R. Castano, M. Judd, R. Anderson, and I. Nenas, "Enabling Autonomous Rover Science through Dynamic Planning and Scheduling," *Proceedings of the 2005 IEEE Aerospace Conference*, Big Sky, Montana, March 2005.

(Fisher, et al., 2002) F. Fisher, T. Estlin, D. Gaines, S. Schaffer, C. Chouinard, R. Knight, "CLEaR: Closed Loop Execution and Recovery – A Framework for Unified Planning and Execution," *Technology and Science IND News Issue 16*, pg. 15-20, September 2002.

(Gat 1991) E. Gat, "Integrating planning and reacting in a heterogeneous asynchronous architecture for mobile robots," in *SIGART Bulletin 2*, 1991, 70-74

(Jonsson, et al., 2000) A. Jonsson, P. Morris, N. Muscettola, K. Rajan, and B. Smith, "Planning in Interplanetary Space: Theory and Practice," *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems*, Breckenridge, CO, April 2000.

(Mishkin, et al., 1998) A. Mishkin, J. Morrison, T. Nguyen, H. Stone, B. Cooper, B. Wilcox, "Experiences with Operations and Autonomy of the Mars Pathfinder Microrover," *Proceedings of the 1998 IEEE Aerospace Conference*, March 21-28 1998, Snowmass at Aspen, Colorado.

(Nenas, et al., 2006) I.A. Nenas, R. Simmons, D. Gaines, C. Kunz, A. Diaz-Calderon, T. Estlin, R. Madison, J. Guineau, M. McHenry, I. Shu, and D. Apfelbaum, "CLARAty: Challenges and Steps Toward Reusable Robotic Software," *International Journal of Advanced Robotic Systems*, 2006.

(Simmons and Apfelbaum, 1998) R. Simmons and D. Apfelbaum, "A Task Description Language for Robot Control," *Proceedings of the Intelligent Robots and Systems Conference*, Vancouver, CA, October 1998.

(Urmson, et al., 2003) C. Urmson, R. Simmons, I. Nenas, "A Generic Framework for Robotic Navigation," *Proceedings of the IEEE Aerospace Conference*, Montana, March 2003.