Appears in the Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)

An Integrated System for Multi-Rover Scientific Exploration

Tara Estlin, Alexander Gray, Tobias Mann, Gregg Rabideau, Rebecca Castaño, Steve Chien and **Eric Mjolsness**
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109-8099
{firstname.lastname}@jpl.nasa.gov

Abstract

This paper describes an integrated system for coordinating multiple rover behavior with the overall goal of collecting planetary surface data. The Multi-Rover Integrated Science Understanding System combines concepts from machine learning with planning and scheduling to perform autonomous scientific exploration by cooperating rovers. The integrated system utilizes a novel machine learning clustering component to analyze science data and direct new science activities. A planning and scheduling system is employed to generate rover plans for achieving science goals and to coordinate activities among rovers. We describe each of these components and discuss some of the key integration issues that arose during development and influenced both system design and performance.

Introduction

Landmark events have recently taken place in the areas of space exploration and planetary rovers. The Mars Pathfinder mission was a major success, not only demonstrating the feasibility of sending rovers to other planets, but displaying the significance of such missions to the scientific community. Future missions are being planned to send additional robotic vehicles to Mars as well as to the outer planets and an asteroid (JPL 1999). In order to increase science return and enable certain types of science activities, future missions will require larger sets of rovers to gather the desired data. These rovers will need to behave in a coordinated fashion where each rover accomplishes a subset of the overall mission goals and shares any acquired information. In addition, it is desirable to have highly autonomous rovers that require little communication with scientists and engineers on Earth to perform their tasks. An autonomous rover will be able to make decisions on its own as to what exact science data should be returned and how to go about the data gathering process.

This paper presents the Multi-Rover Integrated Science Understanding System (MISUS) which provides a framework for autonomously generating and achieving planetary science goals. This system integrates techniques from machine learning with planning and scheduling to enable autonomous multi-rover behavior for analyzing science data, evaluating what new science observations to perform, and deciding what steps should be taken to perform them. These techniques are also integrated with a simulation environment that can model different planetary terrains and science data within a terrain.

Science data analysis in MISUS is performed using machine-learning clustering methods, which use image and spectral mineralogical features to help classify different planetary rock types. These methods look for similarity classes of visible, rock image regions within individual spectral images and across multiple images. Specifically, clustering is performed by a distributed algorithm where each rover alternates between independently performing learning computations using its local data and updating the system-wide model through communication among rovers. Output clusters are used to help evaluate scientific hypotheses and also to prioritize visible surfaces for further observation based on their "scientific interest." As the system builds a model of the rock type distribution, it continuously assembles a new set of observation goals for a team of rovers to collect from different terrain locations. Thus, the clusterer drives the science process by analyzing the current data set and then deciding what new and interesting observations should be made.

A planning and scheduling component is used to determine the necessary rover activities required to achieve science goals requested by the learning system (Rabideau, Estlin, & Chien 1999). Based on an input set of goals and each rover's initial conditions, the planner generates a sequence of activities that satisfy the goals while obeying each of the rover's resource constraints and operation rules. Plans are produced by using an "iterative repair" algorithm which classifies conflicts and resolves them individually by performing one or more plan modifications. Planning is distributed among the individual rovers where each rover is responsible for planning for its own activities. A central planning system is responsible for dividing up the goals among the individual rovers in a fashion that minimizes the total traversing time of all rovers.

The components described above are also integrated with a simulation environment that models multiple-rover science operations in a Mars-like terrain. Different Martian rockscapes are created for use in the simulator by using distributions over rock types, sizes and locations. When science measurements are requested from a terrain during execution, rock and mineral spectral models are used to generate sample spectra based on the type of rock being observed.

The remainder of this paper is organized in the following manner. We begin by characterizing the cooperating rovers application domain and describing our science scenario. Next, we present the MISUS integrated system framework and describe each of its components. We then discuss design decisions and system requirements that arose during integration and any general lessons learned. In the final sections, we discuss related work, planned future work, and present our conclusions.

Cooperating Rovers for Science

Utilizing multiple rovers on planetary science missions has many advantages. First, multiple rovers can collect more data than a single rover. A team of rovers can cover a larger area in a shorter time where science gathering tasks are allocated over the team. Second, multiple rovers can perform tasks that otherwise would not be possible using a single rover. For instance, rovers landed at different locations can cover areas with impassable boundaries. Also, with several rovers, one rover can afford to take more risk and thus attempt tasks that usually might be be avoided. Third, more complicated cooperative tasks can be accomplished, such as taking a wide baseline stereo image (which requires two cameras separated by a certain distance). Finally, multiple rovers can enhance mission success through increased system redundancy. If one rover fails, then its tasks could be quickly taken over by another rover. In all cases, the rovers should behave in a coordinated fashion, dividing goals appropriately among the team and sharing acquired information.

Coordinating multiple distributed agents for a mission to Mars or other planet introduces some interesting new challenges for the supporting technology. Issues arise concerning communication, control and individual on-board capabilities. Many of these design decisions are related, and all of them have an impact on any on-board technologies used for the mission. For example, for an on-board science analysis system, the amount of communication available will determine how much science data can be easily shared. This factor will also affect a planning system by determining how much each rover can coordinate with other rovers to perform tasks. The control scheme will determine which rovers execute what science gathering tasks which affects the on-board components. For instance, some rovers may be utilized only for science data gathering, while other may be used for planning and/or science analysis. Decisions on the on-board capabilities of each rover can also determine the independence of a rover.

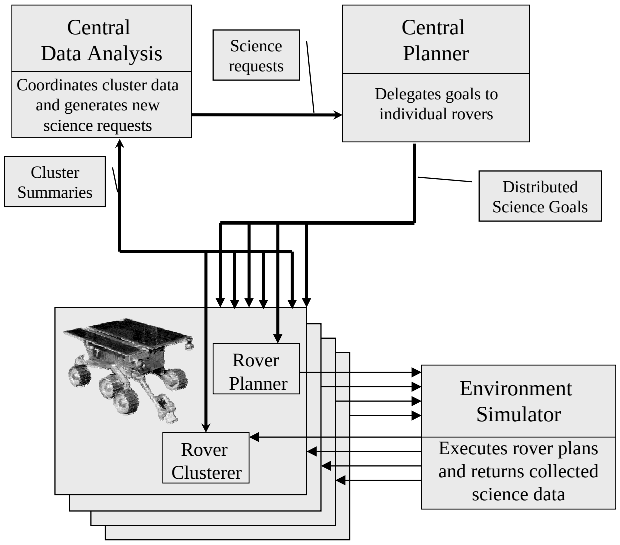

Figure 1: MISUS Architecture Diagram

For the framework presented in this paper, we have initially chosen the configuration of a team of three rovers where each rover has a planning and learning tool on-board. Each rover can thus plan for its assigned goals, collect the required data, and perform science analysis on-board which will direct its future goals. In addition, a central planner and learner are assumed to be located on either a lander or one of the rovers, which are used to coordinate science data and goals.

We evaluate our framework by testing its ability to build a model of the distribution of terrain rocks, classified according to composition as measured by a boresighted spectrometer. To perform testing for different planetary terrain models, different rock fields are generated by using distributions over rock types, sizes, and locations. Science goals consist of requests to take spectral measurements at certain locations or regions. These goals can be prioritized so that if necessary low priority goals can be preempted (e.g. due to resource constraints such as low battery power).

Science goals are divided among the three rovers. Each rover is identical and is assumed to have a spectrometer on-board as well as other resources including a drive motor, a solar panel that provides power for rover activities, and a battery that provides backup power when solar power is not available. The battery can also be recharged using the solar panel when possible. Collected science data is immediately transmitted to the lander where it is stored in memory. The lander can only receive transmissions from one rover at a time.

Multi-Rover Science Architecture

The overall MISUS architecture is shown in Figure 1. The system is comprised of three major components:

- **Data Analysis**: A distributed machine-learning system which performs unsupervised clustering to model the distribution of rock types observed by the rovers. This system is designed to direct rover sensing to continually improve this model of the scientific content of the planetary scene.

- **Planning**: A distributed-planning system that produces rover-operation plans to achieve input rover science goals. Planning is divided between a central planner, which efficiently divides up science goals between rovers, and a distributed set of planners which each plan for operations upon an individual rover.
- **Environment simulator**: A multiple rover simulator that models different geological environments and rover-science operations within them. The simulator manages science data for each environment, tracks rover operations within the terrain, and reflects readings by rover science instruments.

MISUS operates in a closed-loop fashion where the data analysis system can be seen to take the role of the scientist driving the exploration process. Spectra data are received by individual rover clustering algorithms, which attempt to locally model the distribution of rocks according to broad classifications of rock compositions. This information is then sent to a central clusterer which integrates all gathered data into an updated global model and broadcasts the new model back to the distributed clusterers. A prioritization algorithm uses the clustering output to generate a new set of observation goals that will further improve the accuracy of the model. These goals are passed to a central planner which assigns individual rovers to goals in a fashion that will most efficiently serve the requests. Then each rover planner produces a set of actions for that rover which will achieve as many of its assigned goals as possible. These action sequences are sent to the simulator where they are executed and any gathered data is sent back to the rover clusterers. This cycle continues until enough data is gathered to produce distinct clusters for any observed rock types.

In the next few sections, we discuss each of the MISUS system components in more detail.

Data Analysis System

To perform science analysis, we use a machine-learning system which performs unsupervised clustering to model the distribution of rock types in the observed terrain. A primary feature of the MISUS is that the separate rovers cooperate to form a joint consensus for the observed distribution of rock types. Through a learning process, the global distribution model keeps improving as more data is observed over time. For this demonstration prototype, the model used for this distribution is a simple K-means-like unsupervised clustering model, where each cluster represents a different rock type in the sensor space. In the present simulation, each sensor reading is a spectral measurement returning values at 14 wavelengths; learning takes place in the full 14-dimensional continuous space.

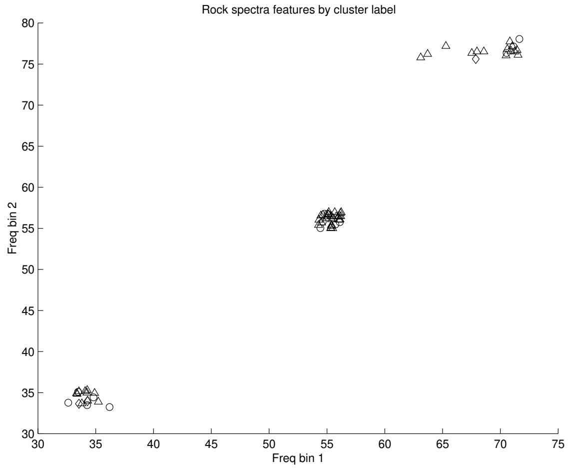

Figure 2: Example spectra feature space

Distributed Clustering At any given time, each rover has a different location on the planetary surface and is sensing different targets. So each rover has its own distinct segment of the overall dataset, stored locally in its data buffer. Over time, each rover collects a new set of data points, or 14-dimensional spectrum readings, adding it to its existing store of data points. Clustering is initiated after each rover has obtained new observations. A sample cluster model (shown for 2 of 14 dimensions) is shown in Figure 2.

Clustering based on the EM (Expectation-Maximization) algorithm, an iterative optimization procedure, normally requires several passes over the entire data. Since rovers must share information through a power-expensive communication channel. Rather than send its local dataset to one or more other rovers, the distributed clustering algorithm allows a rover to send only a small set of parameters which summarizes its local data. Each rover's model parameters are computed locally, then sent to a central clusterer which integrates them into an updated global model (which is also a small set of parameters) and broadcasts that model to all rovers in the system. Each rover takes this global model into account when making its local estimate. This process continues iteratively until convergence. This scheme trades off some accuracy in the global model in order to minimize communication. In the limit of large datasets, this scheme approximates the equivalent non-distributed clustering model (where one processor may examine all the data at once) more and more closely.

The algorithm is very homogeneous, i.e. each processor performs the same computation with the exception of the central clusterer, which performs a few additional computations to compute the global model and broadcast it to the other processors. It also tolerant to processor dropouts, i.e. a circumstance in which one or more rovers contributes zero data to the clustering, for any reason such as a rover malfunction.

$$E = \sum_r^R \sum_n^{N_r} \sum_k^K M_{nk}^r (\|\underline{x}_n - \hat{\underline{\mu}}_k^r\|^2 + \|\hat{\underline{\mu}}_k^r - \underline{\mu}_k\|^2 - \alpha_k)$$
$$+T \sum_r^R \sum_n^{N_r} \sum_k^K M_{nk}^r (\log M_{nk}^r - 1)$$
$$+\sum_r^R \sum_n^{N_r} \lambda_n (\sum_k^K M_{nk}^r - 1)$$

Figure 3: Clustering objective function

Model and Optimization Our clustering model is a special case of the general mixture of Gaussians studied extensively in statistics (Redner & Walker 1984). The K-means clustering model also corresponds to a special form of Gaussian mixture where a hard class membership restriction is made (Bishop 1995). Our model is similar but lifts the restriction.

Hathaway (Hathaway 1986) shows an equivalence between probabilistic and statistical mechanics-style objective functions for mixture distributions. This transformation allows us to generalize the probabilistic objective function to include a temperature variable, allowing us to use deterministic annealing to perform global optimization of the model parameters.

The distributed version of the clustering model follows a development similar to that in (Tsioutsias & Mjolsness 1994) for partitioned neural networks. The entire dataset (across all rovers) $\underline{X}$ contains N vectors $\underline{x} = (x_1, \ldots, x_D)$ indexed by n. Denoting each of the R rovers with an index r, each rover has a subset $\underline{X}_r$ of the data, containing N_r sensor readings. The global, shared clustering model consists of K centroids, $\underline{\mu}_k$. Each rover stores its own local estimates of the centroids, $\hat{\underline{\mu}}_k^r$, based on its subset of the data. M_{nk} denotes the membership of datum $\underline{x}_n$ in cluster k, where $\sum_{k=1}^K M_{nk} = 1$. Cluster membership is determined by a softmax over the distance of a datum to each cluster mean. We will write M_{nk}^r since memberships are only computed and stored locally on each rover.

The clustering algorithm adjusts the values of the centroids in order to minimize the objective function shown in Figure 3, by alternating between a centroid relaxation step, where the cluster means are re-estimated based on the current membership weights, and a membership relaxation step, where the memberships are re-estimated based on the means. α is the reward for committing to cluster k and each λ_n corresponds to a Lagrange multiplier enforcing non-negativity of memberships. T is the temperature parameter.

The first term in this objective function can be identified with minimizing the distance between centroids and the data associated with them and keeping the R estimates of the centroids close to each other. The second serves to prevent negative memberships. The third enforces that memberships sum to unity across the classes.

Goal Selection The clustering model in this initial prototype system may be viewed as the scientific end-product of the exploration. The overall purpose of the system is to increase the accuracy of the clustering model by obtaining sensor readings in regions that are likely to improve the model. An update of the clustering model determines new planetary locations to be explored by the rovers. These locations are sent as formal goals by the learner to the planner.

Recall that clusters are defined in a high-dimensional *spectra space* in which unsupervised learning will identify different rock types. Every datum also has an associated position in physical space, on the planetary surface. Assuming there is some (perhaps very noisy) correspondence between rock type and spatial location, the purpose of goal selection is to direct exploration toward certain rock types by specifying new spatial targets (coordinates in 3-space at which to take sensor readings) according to the observed rock type distribution.

A very simple heuristic for goal selection is used in the current system. A constant number G of new spatial targets will be specified for each cluster. For each cluster, two of the G spatial targets are chosen by first finding the two mutually most distant points (in physical space) of that rock type, then selecting a point in space stochastically from within a neighborhood of each of those 2 points. These goals are given high priority. The rest of the G targets are chosen from neighborhoods of randomly selected rocks in the cluster, and are given lower priority. The idea of this heuristic is to bias the system toward exploration in extremal directions, as well as to explore the rock distribution in a way which balances effort equally between rock types (thus avoiding, say, spending undue energy on a very common rock type at the expense of rare rock types).

Planning System

To produce individual rover plans, we used a distributed version of the ASPEN (Automated Scheduling and Planning Environment) system (Fukanaga *et al.* 1997). ASPEN is a configurable, generic planning/scheduling application framework that can be tailored to specific domains to create conflict-free plans or schedules. Its components include:

- An expressive modeling language to allow the user to naturally define the application domain
- A constraint management system for representing and maintaining domain operability and resource constraints, as well as activity requirements
- A set of search strategies and repair heuristics
- A temporal reasoning system for expressing and maintaining temporal constraints
- A graphical interface for visualizing plans/schedules

ASPEN employs techniques from planning and scheduling to automatically generate the necessary rover activity sequence to achieve the input goals. This sequence is produced by utilizing an iterative repair

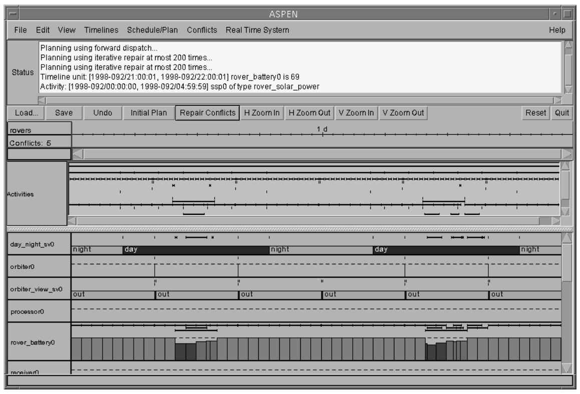

Figure 4: Example ASPEN plan

algorithm (Zweben *et al.* 1994) which classifies conflicts and attacks them each individually. Conflicts occur when a plan constraint has been violated where this constraint could be temporal or involve a resource, state or activity parameter. Conflicts are resolved by performing one or more schedule modifications such as moving, adding, or deleting activities.

A rover that is at the incorrect location for a scheduled science activity is one type of conflict. Resolving this particular conflict involves adding a traverse command to send the rover to the designated site. Other conflicts may include having more than one rover communicating with the lander at a time or having too many activities scheduled for one rover, which over subscribed its power resources. The iterative repair algorithm continues until no conflicts remain in the schedule, or a timeout has expired. Figure 4 shows an example rover-plan displayed in the ASPEN GUI interface.

Distributed Planning To support missions with multiple rovers, we developed a distributed version of ASPEN where it is assumed each rover has an onboard planner. This allows rovers to plan for themselves and/or for other rovers. If communication is slow, it may be useful to have rovers construct their own plans (and to plan dynamically when necessary, which is discussed in future work). Also, by balancing the workload, distributed planning can be helpful when individual computing resources are limited.

The approach to distributed planning utilized in MISUS is to include a planner for each rover, in addition to a central planner. The central planner develops an abstract plan for all rovers, while each rover planner develops a detailed, executable plan for its own activities. The central planner also acts as a router, taking a global set of goals and dividing it up among the rovers. For example, a science goal may request an image of a particular rock without concern for which rover acquires the image. The central planner could assign this goal to the rover that is closest to the rock in order to minimize the traversals of all rovers.

Plan Optimization One of the dominating characteristics of the multi-rover application is rover traversals to designated waypoints. Decisions must be made not only to satisfy the requested goals, but also to provide more optimal schedules. ASPEN can consider optimization goals during the repair process. As certain types of conflicts are resolved, heuristics are used to guide the search towards making decisions that will produce higher quality schedules. In other words, when several options are available for repairing a conflict, these options are ordered based on predictions on how favorable the resulting schedule will be.

For this application, we have implemented heuristics based on techniques from the Multi-Traveling Salesmen Problem (MTSP), which is similar to the Traveling Salesman Problem (TSP) (Johnson & McGeoch 1997). For MTSP, at least one member of a sales team must visit each city such that total traveling time is minimized. Both the central and rover planners utilize the MTSP heuristics. These heuristics are used to select what rover should be assigned a particular science goal and a temporal location for the science activity. In previously reported results, they were shown to make a significant impact in reducing overall traversal distance and expected execution time (Rabideau, Estlin, & Chien 1999).

Environment Simulator

The environment simulator is designed to provide a source of data for the science analysis system by simulating the science gathering activities of the rover. Given the current science scenario, this entails the generation of an environment and the simulation of rover data gathering activities within the environment.

Generation of the environment requires producing a field of rocks for the rovers to traverse. The rock field is generated as a plane with rocks of various sizes embedded at various depths. The simulator maintains information about the mineral composition of each rock, and the spectrum that would correspond to its mineral composition. The size and spatial distributions of the rockfield were developed by examining distributions of rocks observed by the Viking Landers, Mars Lander and Mars Pathfinder. The distribution of minerals that can occur in rocks was developed in collaboration with planetary geologists at JPL, and the spectra associated with rocks are generated from the spectra of the component minerals via a linear-mixing model.

The simulation of the rover activities was done at a coarse level. Such considerations as kinematics and obstacle avoidance were not modeled in this simulation. Other considerations, such as power consumption and memory management were modeled by the planner for plan generation but not simulated by the simulator. The rovers were essentially modeled as roving spectrometers by the simulator. Figure 5 shows several rovers and their spectrometer reaches modeled in a sample rockscape. The simulation of rover activities was accomplished by executing the plan generated by the planner, consisting of a list of movement, rotation, and instrument commands. The simulator would

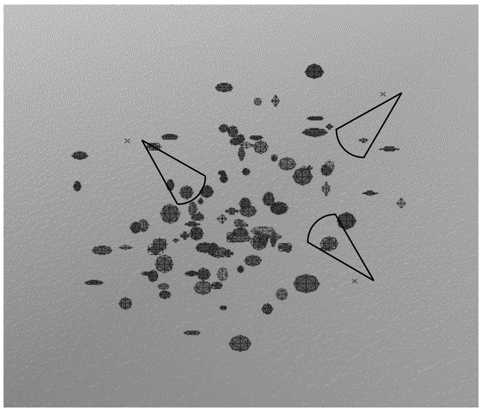

Figure 5: Overhead view of simulated rockscape. Wedges denote different rovers' spectrometers' fields of view.

then, from the location and direction specified by the movement and rotation commands, determine whether or not a rock was visible by the boresighted spectrometer. If so, the simulator would perturb the spectra in an amount proportional to the distance of the rover from the rock in order to simulate instrument noise, and store the spectrum for later communication to the relevant clusterer. After all of the activities in a plan were executed by the simulator (i.e. moves, turns, and data gathering activities), the data was communicated to each clusterer via synchronization agents. The simulator would then wait for the next plan.

Integration Issues

The integration of two AI problem solvers and a simulated environment involved a number of decisions. In this section, we review some of the interesting and challenging issues that arose in performing this integration; we particularly focus on the areas of system and interface design and system performance.

One major integration issue is interfacing between the different components. For instance, the planner was required to produce plans in a format compatible with the action representation required by the simulator. Also, the learner required the ability to ingest any science data returned from the simulator. A more complicated interface arose between the learning and planning components. Issues such as shared representation of goals and objectives had to be resolved.

When specifying a new science goal, the learning component usually requested additional measurements be taken from a particular rock. However, this general request had to be grounded in the form of terrain coordinates in order to represent the goal in the planner's modeling language. In addition, the planning and learning components had to agree on a priority representation that was expressive enough to represent the information required by the learner but that could also be easily utilized during planning to remove goals if necessary due to resource constraints. Another important issue was interfacing between science and engineering representations. Within the planner, constraints may deal with sets of goals, resources and/or states which are primarily scientific, primarily engineering, or which form part of the interface between these two layers.

A separate design consideration was that the interactions between the modules of the integrated system be asynchronous. In other words, each module needed to signal the next module when appropriate, rather than designating one process a control process, which would then control the actions of the others. For instance, the planner would be not begin planning until receiving a new set of science requests from the science analysis module. To that end, we designed a synchronization architecture that would facilitate interprocess signaling and also communication of data. Essentially, each module acted as both a server process and a client process. A process would wait in server mode until the client initiated contact, do its processing, and then initiate contact with the server process of the next system as a client. After the process finished communicating to the next process, it would go back to server mode until it had new data to process.

One important decision is the design of the overall planner, execution, and learner feedback. How often the system loop is run is one issue. Increasing the frequency of feedback improves the responsiveness of the overall system to changes in the inputs (e.g. changes in the observed science data) but increases the computation cost of running the constituent algorithms (e.g. planner, learner). Additionally, due to the design of the algorithms, one may know how much change in information is needed to likely change the results of the computation (e.g. for the learner how much new data is likely to change the collection goals, for the planner how much of a change in execution state or goals is likely to require another plan). While not critically sensitive to the amount of new data it receives, the more data obtained by the science analysis module on a given system cycle, the more its model of the rock type distribution will improve, resulting in useful new exploration goals. If, say, the learner obtains little new information, the targets it decides upon will not be much more useful than those it produced on the last iteration.

A second issue related to system feedback is the length of the horizon (i.e. the allowed plan execution time period) that is considered by each cycle. If this horizon is short, it imposes constraints on how long the cycle must be run (e.g. if the horizon is two hours, the cycle must be run at least every two hours). If the horizon is long, the individual modules may take longer to run (e.g. a planner takes longer to plan for a longer horizon). The number of goals that are requested per iteration also (to some degree) drives the size of the planning horizon since only a certain number of goals can be solved in any set length of time.

The frequency and horizon of each cycle is not constrained by our architecture. However, for our scenario, we chose to have the cycle invoked once per local day and to include a horizon of one day. This time scale is reasonable because science activities are not possible during the night period (as the rover is mainly solar powered) but computation is possible during such periods (using the battery). Thus possible execution time is not expended during planning. However, other choices for cycle frequency and horizon are possible, and may make sense for different mission parameters.

Related Work

The idea of having a scientific discovery system direct future experiments is present in a number of other systems. Work on learning by experimentation, such as IDS (Nordhausen & Langley 1993) and ADEPT (Rajamoney 1990), varied certain quantitative and qualitative values in the domain and then measured the effects of these changes. MISUS differs from these systems in that it interacts with an environment simulator to perform experimentation and it is specialized to particular problems and scenarios in planetary science. MISUS is also integrated with a planning system which constructs the detailed activity sequence needed to perform each experiment based on a domain model.

Other work has used experimentation to learn from the environment but experiments have not been scientifically driven. EXPO (Gil 1993) integrates planning and learning methods to acquire new information by interacting with an external environment. However, while MISUS learns classification models of new geological features, EXPO tries to improve its planning-related domain knowledge.

The distributed clustering presented in this paper bears some similarity to other distributed learning methods, such as (Provost & Hennessy 1999), which are constrained by low-bandwidth communication between processors to share relatively concise data models.

There has also been a significant amount of work on cooperating robots. One related system is GRAMMPS (Bummit & Stentz 1988), which coordinates multiple mobile robots visiting locations in cluttered, partially known environments. GRAMMPS also has a low-level planner on each robot and uses a similar approach to distribute targets, however GRAMMPS does not look at multiple resources or exogenous events. Most other cooperative robotic systems utilize reactive planning techniques (Mataric 1995; Parker 1999). These systems focus on behavioral approaches and do not explicitly reason about assigning goals and planning courses of actions. Furthermore, none of these systems utilize a learning component to drive the system goals.

Future Work

A number of extensions are planned for each component of MISUS. One major extension already under way is to interface with a multiple rover execution architecture (Estlin *et al.* 1999) being developed at JPL that includes a number of additional components including: a real-time multi-rover hardware simulator which models rover kinematics and sensor feedback and control software from the NASA JPL Rocky 7 rover. MISUS is intended to provide the science layer for this architecture, which will allow for more realistic testing of the MISUS framework. In the rest of this section we describe extensions planned for each MISUS component.

Future Work for the Data Analysis System

The learning component described here represents an initial model intended primarily to bring out system issues. The most straightforward of the improvements under consideration are concerned with strengthening the clustering model to include outlier handling, covariance modeling, incremental updating, model size determination, robustness to failure and missing data, and multiple parent representation.

An important area for future work is in goal selection. Combining spectral with spatial distribution models may allow for better-informed targeting. A combined clustering objective function would be more sensitive to novel data in some locations than others. The system could attempt to select target data which maximizes improvement of the data model. This is similar to the saliency measure for an attentive, relaxation-based neural network (Tsioutsias & Mjolsness 1996) To improve the system's applicability to planetary science, we are working to select targets which maximally aid discrimination between two competing hypotheses of geological processes, in collaboration with JPL geologists (Davies *et al.* 1999). One approach is to use stochastic parameterized grammars (Mjolsness 1997) to create a more detailed spatial-spectral model of rock distributions than a mixture of Gaussians.

Future Work for the Planning System

One improvement for the planning component is to enable dynamic planning capabilities. To accomplish this we will utilize a dynamic version of ASPEN, which monitors plan execution and allows re-planning when necessary (Chien *et al.* 1999). To perform autonomous rover-operations, an on-board planning system must be able to respond in a timely fashion to a dynamic, unpredictable environment. Rover plans may often need to be modified due to events such as traverses completing early and setbacks such as failure to reach an observation site.

We also intend to extend the planning model to be more robust to failure situations. For instance, if failure occurs, the planning system should recognize it (e.g. the rover has not responded for a certain amount of time), not send any new goals to that rover, and reassign any current goals assigned to that rover.

Future Work for the Simulator

We are interested in improving the environment simulator by adding different data sources, and by improving the sophistication of the hypotheses investigated.

The visual texture of a rock's surface can give clues to the composition and geological history of the rock, and is a source of information that should be used when attempting to sample from the distribution of rock compositions rather than the spatial distribution of rock locations. We intend to incorporate visual texture as a source of information for the rovers to help them choose a sampling strategy.

Modeling the distribution of rock compositions is a task that can yield useful information for geologists. Consider the scenario where an impact excavates an ancient hydrothermal system, in which there was a stable supply of hot water beneath the surface of Mars at some time in the past. It may be possible to deduce the existence of such a system from study of the impact ejecta scattered on the surface, and examination of the crater interior walls and rim deposits; hydrothermal activity would have altered the mineral characteristics of the excavated deposits.

Conclusions

This paper outlines a framework for coordinating multiple rover behavior in generating and achieving geological science goals. This system integrates techniques from machine learning and planning and scheduling to autonomously analyze and request new science data and generate the action sequences to retrieve that data. We discuss a number of integration issues including developing shared goal and plan representations, coordinating systems asynchronously, and adjusting interface parameters to best serve the overall system goal. We hope the techniques and issues presented in this paper will prove useful to other designers of integrated systems.

Acknowledgments

This work was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. This work was supported by the NASA Advanced Concept Program, Code S, managed by Neville Marzwell, and the NASA Autonomy Program, Code SM, managed by David Atkinson.

The authors acknowledge the invaluable contributions of Ashley Davies for his help in defining relevant geology scenarios for this work and for providing general knowledge of planetary geology. We also thank Padhraic Smyth for providing his environment simulation code, and Rich Maclin for leading the development of the spectra generation process.

References

Bishop, C. M. 1995. *Neural Networks for Pattern Recognition.* New York, NY: Oxford Univ. Press.

Bummit, B., and Stentz, A. 1988. GRAMMPS: A generalized mission planner for multiple mobile robots in unstructured environments. In *Proceedings of the IEEE Conference on Robots and Automation.*

Chien, S.; Knight, R.; Stechert, A.; Sherwood, R.; and Rabideau, G. 1999. Integrated planning and execution for autonomous spacecraft. In *Proceedings of the 1999 IEEE Aerospace Conference.*

Davies, A.; Mjolsness, E.; Gray, A.; Mann, T.; Castano, R.; Estlin, T.; and Saunders, R. 1999. Hypothesis-driven active data analysis of geological phenomena using semi-autonomous rovers: exploring simulations of martian hydrothermal deposits. In *American Geophysical Union Spring Meeting.*

Estlin, T.; Hayati, S.; *et al.* 1999. An integrated architecture for cooperating rovers. In *Proceedings of the 1999 International Symposium on Artficial Intelligence, Robotics and Automation for Space.*

Fukanaga, A.; Rabideau, G.; Chien, S.; and Yan, D. 1997. Towards an application framework for automated planning and scheduling. In *Proceedings of the 1997 International Symposium on Artficial Intelligence, Robotics and Automation for Space.*

Gil, Y. 1993. Efficient domain-independent experimentation. In *Proceedings of the Tenth International Conference on Machine Learning*, 128–134.

Hathaway, R. J. 1986. Another interpretation of the EM algorithm for mixture distributions. *Statistics and Probability Letters* 4:53–56.

Johnson, D., and McGeoch, L. 1997. The traveling salesman problem: A case study in local optimization. In Aarts, E. H. L., and Lenstra, J. K., eds., *Local Search in Combinatorial Optimization.* London: John Wiley and Sons. 215–310.

JPL 1999. *http://www.jpl.nasa.gov/missions/.*

Mataric, M. 1995. Designing and understanding adaptive group behavior. *Adaptive Behavior* 4(1):51–80.

Mjolsness, E. 1997. Symbolic neural networks derived from stochastic grammar domain models. In Sun, R., and Alexandre, F., eds., *Connectionist Symbolic Integration.* Lawrence Erlbaum Associates.

Nordhausen, B., and Langley, P. 1993. An integrated framework for empirical discovery. *Machine Learning* 12:17–47.

Parker, L. E. 1999. Cooperative robotics for multi-target observation. *Intelligent Automation and Soft Computing* 5(1):5–19.

Provost, F. J., and Hennessy, D. N. 1999. Scaling up: Distributed machine learning with cooperation. In *Proceedings of the Thirteenth National Conference on Ariticial Intelligence.*

Rabideau, G.; Estlin, T.; and Chien, S. 1999. Working together: Automatic generation of command sequences for multiple cooperating rovers. In *Proceedings of the 1999 IEEE Aerospace Conference.*

Rajamoney, S. 1990. A computational approach to theory revision. In Shrager, J., and Langley, P., eds., *Computational Models of Scientific Discovery and Theory Formation.* San Mateo, CA: Morgan Kaufman. 225–254.

Redner, R. A., and Walker, H. F. 1984. Mixture densities, maximum likelihood, and the EM algorithm. *SIAM Review* 26:195–239.

Tsioutsias, D., and Mjolsness, E. 1994. Optimization dynamics for partitioned neural networks. *International Journal of Neural Systems* 5(4).

Tsioutsias, D., and Mjolsness, E. 1996. A multiscale attentional framework for relaxation neural networks. *Advances in Neural Information Processing Systems* 8.

Zweben, M.; Daun, B.; Davis, E.; and Deale, M. 1994. Scheduling and rescheduling with iterative repair. In Zweben, M., and Fox, M., eds., *Intelligent Scheduling.* San Francisco, CA: Morgan Kaufmann. 241–256.