AUTONOMY SOFTWARE ARCHITECTURE FOR LORAX (LIFE ON ICE ROBOTIC ANTARCTIC EXPLORER)

Ari K. Jónsson[(2)], ConorMcGann[(2)], Liam Pedersen[(2)]), Michael Iatauro[(2)]), Srikanth Rajagopalan[(2)]

[(1)]Research Institute for Advanced Computer Science, NASA Ames Research Center, Mailstop 269-2, Moffett Field, CA 94035, USA, Email: jonsson@email.arc.nasa.gov
[(2)]QSS Group Inc., NASA Ames Research Center, Mailstop 269-2, Moffett Field, CA 94035, Email: {cmcgann, pedersen, miatauro, srikanth}@email.arc.nasa.gov

ABSTRACT

The objective of the LORAX mission is for an autonomous rover to navigate more than 100 kilometers on the Antarctic ice, while taking ice samples and analyzing them for evidence of life. The mission will provide important scientific results about life in Antarctica, while also evaluating technology for future Mars missions.

The LORAX rover operations will be entirely autonomous, requiring a reliable autonomous control system that can handle environmental uncertainties and changes in operations goals. In this paper, we describe a new autonomous rover control architecture, which was developed for LORAX as part of a mission and vehicle design study. The key notions in the architecture are the use of a flexible plan representation and continuous replanning to handle environmental uncertainty. The autonomy software will also be used early in the mission design process, to evaluate choices such as battery capacity, by examining expected outcomes of operations in varying conditions.

1. INTRODUCTION

LORAX is a robotic astrobiological study of the ice field surrounding the Carapace Nunatak near the Allan Hills in Antarctica. The study comprises a 30 km traverse, followed by a 100km traverse. On each traverse the ice will be sampled at over 100 sites to survey microbial ecology and to record environmental parameters.

Numerous factors drive the need for autonomy in the LORAX mission. First, it will demonstrate robotic science technologies applicable to future Mars mission, and as such, must operate with limited human oversight and interaction. Secondly, the LORAX science goals require minimizing biological contamination risk from human presence. Finally, the mission takes place in a highly uncertain and dynamic environment with limited resources, requiring the rover to adapt its plan of action to ensure successful mission completion, while maximizing the number of samples analyzed.

The autonomy requirements from LORAX are shared by many robotic exploration tasks. Consequently, the LORAX autonomy architecture is a general architecture for on-board planning and execution in environments where science return is to be maximized against resource limitations and other constraints. Three key elements set it apart from other general planning-execution architectures used for rover operations:

1. Flexible plans describe families of plans having the same structure and outcomes. This flexibility increases the applicability of a plan in changing environments and reduces need for replanning due to minor variations.

2. Continuous replanning to do on-line plan optimization. This allows the autonomy system to seamlessly modify plans in response to outcomes that differ from what was expected.

3. Resource envelopes bound expected resource profiles. These envelopes provide better information on the feasibility of candidate plans in flexible and uncertain situations, and offer better search control information during planning.

The backbone of the LORAX autonomy architecture is EUROPA, a constraint-based planning framework. EUROPA supports the description of temporal actions and states, complex constraints and operations rules, and complex resources. The underlying representation and reasoning is based on dynamic constraint network reasoning. The advantages of the constraint-based approach include flexibility, generality, and improved efficiency.

The EUROPA system uses new methods to bound resource profiles, making it possible to reason effectively about resource use in partial and flexible plans. The LORAX autonomy architecture extends the baseline technology to handle certain types of non-

Proc. of 'The 8th International Symposium on Artifical Intelligence, Robotics and Automation in Space - iSAIRAS', Munich, Germany. 5-8 September 2005, (ESA SP-603, August 2005)

linear and time-dependent resources, such as internal rover temperature and battery capacity.

Finally, the LORAX autonomy architecture defines a planning and execution system that supports parallel replanning and execution. Execution results and real-world outcomes are used to generate alternative plans, which then, if appropriate, are spliced in to replace the current plan. This is made possible by the use of flexible plans, and the propagation of information through constraint reasoning.

The LORAX project is work-in-progress. The rover hardware design is being finalized, and an initial prototype of the autonomy architecture is being evaluated. A full version of the autonomy software will be ready for operations tests on the rover in early 2006. This will be followed by operations tests on glaciers, and then two missions to the Antarctic in 2007 and 2008.

2. THE LORAX MISSION

The primary science objective of the LORAX mission is to develop a map of the concentration and type of organic material over a large varied area on the Antarctic ice sheet. At the same time, robotic exploration of the Antarctic ice sheet provides an analogue of possible future search for evidence of life on other planetary surfaces in the solar system, most notably in the polar regions on Mars.

The geographic target of the LORAX mission is the area around the Carapace Nunatak in Antarctica. In this area, ice flows and prevailing winds meet at right angles, providing a variety of surface and weather conditions, which impact how much and what kind of microbial organism is found in the ice. The rover will circumnavigate two nunataks (which are mountains that rise up through the ice sheet), in two separate efforts, with the first traverse being 30 km, and the second 100 km. On each traverse, the rover will pass through windward and leeward areas, upstream and downstream ice flows, as well as hard-packed snow and blue ice. At regular intervals, the rover will collect ice samples and analyze them. The sampling system will take cores at three different depths, and bring the samples to the analysis system. The analysis system uses an ultra-violet spectrograph to estimate the concentration of microbial life in each sample, and to estimate the types of microbial life found.

The role of the rover is crucial in this mission. The traverse and sampling must be done without risking contamination from human operators, as even small amounts of microbial contamination will significantly skew sample results. Other sources of biological material, in particular hydro-carbons, must also be avoided, necessitating the use of clean power sources. Finally, to demonstrate the applicability of the technology to future deep space missions, the LORAX mission must be operated in a manner consistent with off-planet operations.

The LORAX rover is based on the NOMAD rover [1], and inherits the basic drive and navigation systems. A prototype of the LORAX rover is shown in Fig.1. To provide clean renewable energy, the rover will use solar panels and a wind turbine to generate electricity. To handle periods without significant sunlight or wind, the rover will have a battery with sufficient capacity to keep it alive for a number of days. The primary science tool will be a combination of a sample acquisition system and an ultra-violet spectrographic analyzer. The acquisition system is an arm that can collect a suitable sample of ice and snow, and bring it to the analyzer. The analyzer then crushes the sample and spreads it out in an analysis box. A moveable spectrographic sensor maps the distribution of microbial life in the sample. The same sensor can then also map the distribution of different types of microbial organisms, such as photosynthesizing life, in the sample. Once the sample has been analyzed, it is ejected from the system.

Fig. 1. The LORAX rover prototype

The expected baseline is to collect at least 100 evenly spaced samples on each circumnavigation. However, unexpectedly high variations in the gradient between samples might lead to additional samples, or re-sampling. At the same time, the mission will be run within a tight time limit, due to weather conditions in Antarctica. If weather or operational delays preclude the completion of the traverse and the collection of all 100 samples, the number of samples will be reduced, in order to complete the circumnavigation goals.

3. LORAX AUTONOMY REQUIREMENTS

The need for autonomy in the LORAX mission is driven by a number of factors. As an analogue for future deep space missions, the mission cannot rely on direct human operations, be it via physical presence or tele-operations. As noted above, the mission science goals are also highly sensitive to human interference, due to danger of contamination. Finally, the mission operates in a highly dynamic environment with a tight deadline for completion. Changes in the environment and rover operations results are likely to frequently invalidate given plans, forcing replanning or plan adjustments.

In comparison to other autonomous rover operations, LORAX is in some ways simpler and in other ways more complex. Navigation, which is often one of the key complications in rover autonomy, is expected to be fairly straightforward. The rover will largely follow a pre-determined route to circumnavigate a nunatak, and significant obstacles are expected to be rare. However, the strict power and thermal constraints, the impact of dynamic weather on energy levels and thermal states, and the constant trade-off between sampling and completing the circumnavigation give rise to a more complex dynamic optimization problem.

At a high level, the goals of the autonomous control software are straightforward. Scientists will specify a set of goals, and a method for trading off goals as needed. For example, the initial goals may be for the rover to traverse a given 100 km path, taking and analyzing samples at 100 evenly spaced locations, where each location involves three samples, each analyzed in four different ways, with two cleaning cycles after each location and one cleaning cycle between sample layers. The given trade-off strategy will be to do as many locations as possible, but the circumnavigation goal has priority. Throughout the mission, the goals may change, as scientists adjust their strategy for sample analysis at each location, and as new location sample requests are added, possibly in response to unexpectedly large changes in analysis results.

The autonomous system manages all systems on board the rover, and is responsible for maintaining safe energy levels, safe thermal states, as well as to abide by given operations constraints, while maximizing the expected value gained from the remainder of the mission.

Briefly, the key elements of the LORAX autonomy requirements are as follows:

Objective maximization: The problem is an optimization problem – maximizing the expected outcome from the remainder of the mission. In simplified terms, this means achieving the circumnavigation goal while maximizing the number of samples. However, additional complexities arise from uncertainty and affected resources, such as power and thermal states. Consequently, the safety margin, or the likelihood of violating a critical constraint, must be factored into the objective.

Dynamic environment: The environment is highly dynamic. Most notably, the weather greatly impacts rover operations, and in particular, the power and thermal states. Forecast information will be available, but clearly forecast will never be fully accurate, and become less and less reliable as they look further into the future. In addition, there are dynamic elements to rover operations, in particular the drive times and the time needed for sample analysis, which depends on how much organic material is in a given sample.

Temporal planning: Time plays a key role in the problem. All actions are durative, and few states are steady states that can be held indefinitely. This means that many limitations and expectations are specified in terms of temporal constraints.

Complex resources: Power and thermal states are complex functions and have a critical impact on operations. The state values are impacted by many types of activities, and also vary over time, without any actions. Furthermore, both thermal and power states may require planned actions to be postponed, modified or cancelled, and for new actions to be added into the plan, to avoid depleting the battery or freezing the rover systems.

Execution and diagnosis: While the primary emphasis of this work is on the higher-level reasoning element, a robust execution component must be provided. Actions may fail for a variety of reasons, necessitating appropriate responses. This involves identifying that an action has failed, identify a reason if possible, possibly attempt a recovery action, and in the worst case, enter a safe state and request further instructions from engineers.

4. THE EUROPA PLANNING FRAMEWORK

The LORAX autonomy architecture is built on a constraint-based plan representation and reasoning framework. In constraint-based planning, actions and states are durative, holding over intervals of time.

Each such interval is described by variables representing the start and end times, and the various parameters of the action or state in question. The interval variables are connected by constraints, which specify the legal value combinations for variables. For example, a duration limit is specified as a constraint on the start and end variables of an interval. If the duration depends on action or state parameters, then the variables representing those parameters are also involved in the constraint.

The specific constraint-based planning framework used in LORAX is called EUROPA (Extendible Uniform Remote Operations Planning Architecture) [1]. Domain models specify interval types, i.e., action and state types, along with configuration rules, which generalize the notion of preconditions and effects. In simple terms, a configuration rule consists of an interval pattern, and an or-and statement about the existence of other intervals and how they must be related. To satisfy a configuration rule, for any interval that matches the given pattern, there must exist suitable "support" intervals, satisfying necessary relations, so that the logical or-and statement is true. For example, a very simplistic domain might define three action and state types: **at**(*loc*), **fly**(*loc*,*loc*), and **drive**(*loc*,*loc*), where *loc* parameters take on values from a set of defined locations. Durative actions and states are then described as holding over intervals using the syntax holds(**P**,*s*,*e*) to indicate that predicate **P** holds between times *s* and *e*. For example, holds(**at**(*x*),*s*,*e*) means that **at**(*x*) is true between *s* and *e*. Note that since *x*,*s* and *e* are variables, this statement can be made even when the location and start and end times are not yet known. A simple example configuration rule for this domain would then be:

$$\text{holds}(\mathbf{at}(y),s,e):$$
$$(\text{holds}(\text{fly}(x_1,y_1),s_1,e_1) \wedge y=y_1 \wedge e_1=s)$$
$$\vee \quad (\text{holds}(\text{drive}(x_2,y_2),s_2,e_2) \wedge y=y_2 \wedge e_2=s)$$

meaning that in order for **at**(*y*) to start holding at time *s*, there must have been either a **fly** or a **drive** action with destination *y* that ended at time s_1, which is equal to *s*.

In EUROPA, any candidate plan gives rise to an underlying constraint network. Constraint propagation and reasoning techniques are used to determine the consistency of the network, and to derive conclusions about variable values being excluded from consideration due to combination of constraints. To build a valid plan, a search mechanism is used to modify a candidate plan until a valid complete plan is found. Between modification steps, the constraint reasoning methods can be used to exclude invalid options and to identify inconsistent candidate plans that cannot be extended to complete valid plans.

The constraint-based planning framework is a natural fit for the LORAX autonomy architecture, providing temporal reasoning with durative actions and states that expire. Recent extensions of the EUROPA framework have added the capability to reason about complex resources for flexible temporal plans. This new capability provides the foundation for reasoning about power and thermal resources for LORAX.

The resource reasoning in EUROPA is based on bounding resource envelopes for flexible temporal plans. The mechanism involved is a variation of the configuration rules, but the consequences describe resource events that cause a change in resource levels. The result is that for any given candidate plan, there is a set of partially ordered resource events. Resource bounding methods, such as those described in [4], and specifically the approach defined in [0], can efficiently calculate bounds on possible resource values, covering all possible plan instantiations. This can be used to ensure that a flexible plan can be safely executed, or to identify candidate plans that cannot be made into legal plans.

5. LORAX AUTONOMY ARCHITECTURE

The LORAX autonomy architecture is in many ways similar to traditional rover control architectures. There is a high-level planning element, an execution element, which includes a health monitor and recovery capability, and a functional system interface element. What sets the LORAX architecture apart from other rover systems is that the high-level planning layer has on-line optimization capabilities, and the tight connection between the planning and the execution layers.

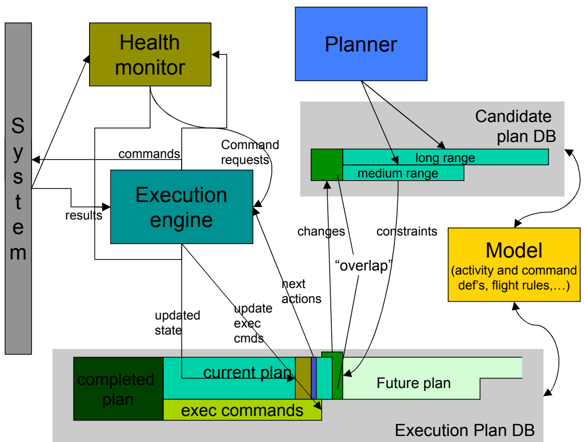

Fig. 2. LORAX autonomy software diagram

The architecture is shown in Fig. 2. There are three main functional components: the high-level planner, the health monitor and the execution engine. There are two plans in the system at any time; a plan being executed (the active plan), and a candidate future plan (the plan being built). These two plans are based on the same domain model, meaning that they share action and state types, and configuration rules. However, different levels of the plans may be at different levels of abstraction.

EUROPA domain models can naturally represent abstractions by having action and state types that expand into more detailed actions and states. These expansions are described as configuration rules. In addition, approximations of resources and other domain constraints can be represented for higher levels of abstractions, as long as those approximations are sound, i.e., don't exclude valid plans.

This capability allows candidate and active plans to be represented at different levels in the LORAX software. A candidate plan will, at a very high level, be completed to end of mission, thus ensuring that the overall mission goals are being taken into considerations when decisions are being made. A candidate plan will then also have a more detailed plan for the near future, thus taking into account information available about near-term expectations, in particular, the weather. An active plan will have the same levels of abstractions as a candidate plan, as well as an execution level, where rover commands are specified.

The high-level planning system will, as in most rover control systems, generate candidate plans for the future, which then get turned into active plans for execution. A key requirement for the LORAX planner is that it be an optimizing planner; generating plans that maximize the estimated expected value, in terms of achieving mission goals. As noted above, the plan evaluation for this optimization involves not only the core science goals in the plan, but also the robustness of the plan to dynamic outcomes.

The planning system will also continually work on alternative candidate plans, using the most recent information about resource levels and execution outcomes. If a plan is found that is evaluated to be notably better than the current plan, then the new plan is spliced into the active plan, replacing most of the current plan. Needless to say, this splicing must be done at a point that is not yet being executed, which in turn means that alternative candidate plans will have a prefix that is common with the current plan. This on-line replanning is a key element of dealing with the highly dynamic and often uncertain environment in which the LORAX rover will operate.

The execution system will traverse the active plan, and expand current actions to rover commands that then get sent to the rover systems. The expansion of actions into commands is defined in a manner similar to the configuration rules that specify other expansions and safety rules. However, the expressiveness of the execution expansion rules will be limited, so as to ensure that no unbounded search is being done. The notion of using the same kind of domain description for planning and execution is part of the IDEA architecture [5]. In fact, a version of IDEA will likely be used to implement the execution engine.

The health monitor and other support modules are similar to those found in traditional systems. However, their interfaces will be defined in such a manner that information is mapped into elements of constraint-based plans, i.e., states, events or actions.

6. PROTOTYPE FOR MISSION SIMULATION

For LORAX, the close interaction of weather and operations on mission-critical resources of heat and energy make competent reasoning about resources essential and challenging. To ensure the selected technology could meet the requirements of this dynamic domain, and to feed into the system design effort, it was decided to develop a prototype for mission simulation. The product of this work could then be applied to evaluate alternate system designs, with the particular goal of identifying appropriate power generation and storage specifications.

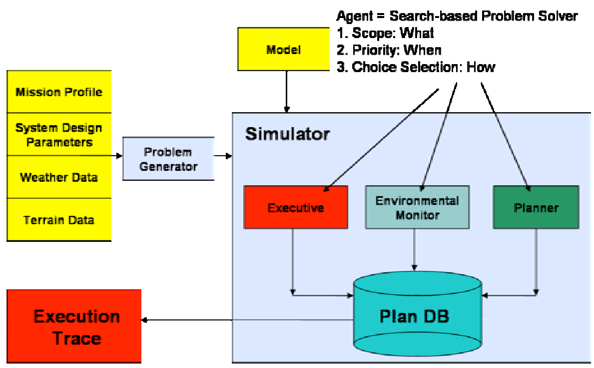

Fig. 3. A schematic of the prototype mission software and how it is used to provide simulation-based evaluation of design alternatives.

The prototype focused on the main functional areas of mobility control and instrument management within the context of a resource-constrained robot in a dynamic environment. The simulator system design is depicted in Fig. 3.

The inputs are: 1) a domain model describing the mission systems, sub-systems and their interactions; 2) a mission profile consisting of a temporally ordered sequence of sampling goals; 3) a set of system design parameters used to explore design variations primarily in terms of battery, wind-turbine and solar-panel specifications; 4) wind and solar data taken from on-site measurements; 5) a terrain data set indicating elevation and terrain characteristics for a known path. Input data is translated into a problem specification and fed into the simulator. The output of the system is a trace of simulated execution, providing temporally qualified values for state and action throughout the mission.

The simulator consists of: 1) a plan database to store and manage the past, current and future states and actions as simulation proceeds; 2) a planner to turn mission goals into executable plans; 3) an environment monitor to provide periodic update of actual and expected sensor data; 4) an executive responsible for scheduling commands for execution according to the plan as the simulation clock progresses. The planner, environment monitor and executive are distinct configurations of a problem-solving framework provided in EUROPA. The basic operation of the simulator is that at each clock tick the planner, execution monitor and executive are sequentially invoked to update the plan database. The planner is configured to conduct task decomposition and refinement to meet goals, over a full mission planning horizon. The environment monitor is configured to bind weather parameters for the current clock time based on real weather data. Note that this information is only made available at the point of execution so that it does not provide any foresight to the planner that would be absent in real life. The environment monitor also binds certain operational parameters (e.g. the end times of uncontrollable activities) via random selection within a given range. The executive makes scheduling decisions to start and stop controllable activities at the current clock-tick based on current and future resource levels.

The simulator prototype currently provides a detailed simulation for a 24 hour mission cycle. The model handles dynamic sample acquisition and analysis allowing for variations in the number of cleaning cycles required and the type of scanning. It models the energy production from wind and solar sources. It models energy consumption for basic system operation, as well as task-driven activities such as driving and sampling. Furthermore, the thermal impact of external temperature and heat dissipation due to system operation are integrated to compute resource envelopes for heat and temperature which are employed in heuristics for decision making and as constraints on operation.

7. FUTURE WORK

The current status of the simulator demonstrates the representational sufficiency of the model-based planning technology and the technical feasibility of a planning–based simulation system. However, in order to deliver the expected benefits we must address the following extensions:

1. Simulator completion - while the basic components required for simulation are now in place, we must complete the model development to accurately reflect the energy and heat production and consumption formulae currently encoded and available as MatLab functions. This will yield greater accuracy for the simulator. Currently the simulation operates on a scaled down version of the mission (over 24 hours instead of 30 days) to expedite testing. Once the model is complete we will extend simulation runs for full missions.

2. Develop an optimization algorithm - the simulator is designed currently to find any plan that will accomplish mission objectives given the input system specifications. It does not yet seek optimal plans. As noted above, mission decisions will be made in a way that maximizes the expected value in terms of science goal achievements. Taking this optimization into account for scenario evaluation can help avoid overly pessimistic expectations in terms of outcomes.

3. Conduct simulation-based system evaluation - we will apply the simulator to help determine certain aspects of the rover design. A prime example is determining the battery capacity needed to reliably accomplish the mission while avoiding having unnecessarily large batteries. To accomplish this, a variety of simulations will be run for particular system designs, and a range of design parameters will be explored to find the set that work best.

The simulator is important in that it drives early technology feasibility and development, and it offers the chance of leveraging the autonomy software during the system design process. However, it is not the end product. Rather, the components of the simulator, notably the planner, the executive, the model, and the plan database will be integrated to provide the on-board autonomy architecture described in Section 5. The main requirements to accomplish this are to: 1) integrate actual system monitoring to replace the role of the environment monitor; 2) integrate a health monitor which can raise events for failures occurring; 3) integrate hooks for invocation of system control api's for command execution (as updates are made by the executive; 4) conduct field tests for the autonomy software for nominal and off-nominal execution paths.

Given the incremental nature of the development process for the project overall, it will be important to continually evolve the domain model to adapt to alterations in component and system behaviours.

8. CONCLUDING REMARKS

The LORAX architecture is closely related to what is being used in LITA (Life In The Atacama desert). That rover control system uses the IDEA architecture, which is in turn implemented on top of the EUROPA framework. In IDEA, the execution of plans is viewed as a reactive, short-term planning problem, allowing for more natural, more compact, and more powerful sets of rules for executing plans. As for any such approach, there is some concern about providing guaranteed response times when execution steps may lead to exhaustive and expensive search for plans. However, in LORAX, the expressivness of the execution expansions are limited, ensuring a smooth expansion and timely execution. Furthermore, in LITA, a variety of heterogeneous components are integrated to accomplish the overall autonomy capabilities. In addition to the IDEA executive, TEMPEST [8] is used for on-board planning, and TDL [7] is used for instrument management. In contrast, LORAX employs a single model and supporting plan database for all aspects of autonomous control, offering a simpler development and deployment architecture which should prove less error-prone in integration

Another closely related planning and execution system is CASPER [2]. In CASPER, the generated plans are fully instantiated, meaning that deviations from expected outcomes and environmental impacts can more easily make the plan incorrect. When that occurs, the high-level planner gets invoked, to repair the plan flaw found. The core notion of this repair is similar to what is in the LORAX architecture, although in LORAX, the repairs are done pre-emptively when possible, and the use of flexible plans reduces the number of expected execution failures. The trade-off is that calculating resource levels for a flexible plans is significantly more complex than it is for fully instantiated plans.

The LORAX mission pushes the capabilities of on-board rover control software in a number of ways. The environment is highly dynamic and changes have significant impact on plan execution success and how mission goals are best achieved. The critical resources, thermal and energy states, are more complex than what current systems handle. The mission goals are given in terms of an optimization problem, which must be solved on the fly on board the rover. At the same time, the traditional focal points of rover control software, navigation and localization, are fairly straightforward for LORAX.

The architecture described here is not unlike some existing software systems and architectures. However, it does identify a number of novel approaches for solving the above-mentioned problems. In this paper, we have provided a glimpse into the ongoing work on the LORAX autonomy framework, but a great deal of work remains to be done. As work progresses, more concrete definitions of the architecture and details of implemented algorithms will be forthcoming.

9. ACKNOWLEDGEMENTS

Funding for this work was provided by NASA's Astrobiology and Science for Exploring Planets (ASTEP) program, with additional funds from NASA's Intelligent Systems (IS) program.

10. REFERENCES

1. D. Apostolopoulos, M.D. Wagner, B.N. Shamah, L. Pedersen, K. Shillcutt, W.L. Whittaker, "Technology and Field Demonstration of Robotic Search for Antarctic Meteorites" in *International Journal of Robotic Research*, special Field & Service Robotics issue, November 2000.

2. T. Estlin, G. Rabideau, D. Mutz, and S. Chien. Using Continuous Planning Techniques to Coordinate Multiple Rovers. In IJCAI Workshop on Scheduling and Planning, Stockholm, Sweden, August 1999.

3. J. Frank and A. Jónsson. Constraint-Based Interval and Attribute Planning. *Journal of Constraints Special Issue on Constraints and Planning*. October, 2003. Volume 8. Number 4.

4. P. Laborie, "Algorithms for propagating resource constraints in AI planning and scheduling: existing approaches and new results", *Artificial Intelligence*, Volume 143 , Issue 2 (February 2003) 151 - 188

5. N. Muscettola, G. A. Dorais, C. Fry, R. Levinson, and C. Plaunt, "IDEA: Planning at the core of autonomous reactive agents," in Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space, October 2002.

6. N. Muscettola: "Incremental Maximum Flows for Fast Envelope Computation". Proceedings of the 13th International Conference on Automated Planning and Scheduling, 2004: 260-269

7. R. Simmons and D. Apfelbaum. "A Task Description Language for Robot Control", In Proceedings of Conference on Intelligent Robotics and Systems, Vancouver, 1998

8. P. Tompkins, A. Stentz, and W.L. Whittaker, "Mission-Level Path Planning for Rover Exploration," Proceedings of the 8th Conference on Intelligent Autonomous Systems (IAS-8), March, 2004.