Mission Planning and Target Tracking for Autonomous Instrument Placement[1]

Liam Pedersen [1], David E. Smith[2], Matthew Deans[1], Randy Sargent[1], Clay Kunz[1], David Lees[1], Srikanth Rajagopalan[1]
[1]QSS Group, Inc
[2]NASA Ames Research Center
Moffett Field, CA 94035-1000
650 604-0829
[1]{pedersen, deano, rsargent, clay, lees, srikanth}@email.arc.nasa.gov
[2]David.E.Smith@nasa.gov

Abstract—Future planetary rover missions, such as the upcoming Mars Science Laboratory, will require rovers to autonomously navigate to science targets specified from up to 10 meters away, and to place instruments against these targets with up to 1 centimeter precision. The current state of the art, demonstrated by the Mars Exploration Rover (MER) mission, typically requires three sols (Martian days) for approach and placement, with several communication cycles between the rovers and ground operations. The capability for goal level commanding of a rover to visit multiple science targets in a single sol represents a tenfold increase in productivity, and decreases daily operations costs.

Such a capability requires a high degree of robotic autonomy: visual target tracking and navigation for the rover to approach the targets, mission planning for determining the most beneficial course of action given a large set of desired goals in the face of uncertainty, and robust execution for coping with variations in time and power consumption, as well as the possibility of failures in tracking or navigation due to occlusion or unexpected obstacles.

We have developed a system that provides these features. The system uses a vision-based target tracker that recovers the 6-DOF transformations between the rover and the tracked targets as the rover moves, and an off-board planner that creates plans that are carried out on an on-board robust executive. The tracker comprises a feature based approach that tracks a set of interest points in 3-D using stereo, with a shape based approach that registers dense 3-D meshes. The off-board planner, in addition to generating a primary activity sequence, creates a large set of contingent, or alternate plans to deal with anticipated failures in tracking and the uncertainty in resource consumption.

This paper describes our tracking and planning systems, including the results of experiments carried out using the K9 rover. These systems are part of a larger effort, which includes tools for target specification in 3-D, ground-based simulation and plan verification, round-trip data tracking, rover software and hardware, and scientific visualization. The complete system has been shown to provide the capability of multiple instrument placements on rocks within a 10 meter radius, all within a single command cycle.

TABLE OF CONTENTS

[1] IEEEAC paper #1243, Version 2, Updated December 10, 2004

1. INTRODUCTION

"*… It takes the MER rover a day to do what a field geologist can do in about 45 seconds...*"[2] The MER vehicles currently on Mars can require up to three sols to approach a distant target and accurately place an instrument against it. With MER rover operations costing $4M-$4.5M per day and requiring 240 operators working 24/7 during the nominal mission, speeding this up and reducing operator workload has a large potential to save costs and increase productivity.

Our goals are inspired by early design requirements for the 2009 Mars Science Laboratory rover [1]: single cycle instrument deployment against a single 10m distant target with 1 cm precision. However, we have taken this requirement a step further: autonomous instrument placement on multiple science targets in a single command cycle. This represents a tenfold increase in capability compared to MER.

Achieving this goal of single cycle activity commanding for planetary rovers requires broad advances in robotics, autonomy and user interfaces. Namely:

Target tracking and navigation -- Localization errors from rover odometry and deduced reckoning are too large to guide a rover to a 10m target with the required accuracy. Therefore, the rover must explicitly track target locations as it navigates about the worksite and avoids obstacles. Because features are selected for scientific relevance, they are not necessarily those features which best facilitate visual tracking. The rover might move completely around targets. Lighting changes are common, due to changing sun angles or rover shadowing.

Automated instrument placement – the chosen placement point on a target rock may potentially harm the instrument. Even assuming that users can be certain from 10m away that a point is safe, accumulated designation, tracking and placement errors mean instrument placement could still be unsafe. The rover must therefore autonomously confirm the safety of a presumed target point, and find alternatives if it is not.

Activity planning and execution – Going to multiple targets implies significant time and energy expenditure coupled with great uncertainty in their usage. Planetary rovers have very tight constraints on these resources that must be enforced. In addition, target tracking imposes constraints on the paths a rover can take (Figure 6), which targets it can go to and in what order. Violating these imposes a risk that the targets will be lost.

Rapid activity specification and data interpretation tools -- Ground data systems are necessary for users to rapidly identify, prioritize and specify many potential targets, evaluate the plan of action, and understand the data returned from the multiple samples the rover actually visited.

The next sections of this paper describe the mission scenario and assumptions underlying our system; an overview of the technology components addressing the functions identified in the mission scenario and how they fit together; a detailed technical description of the activity planning, execution, navigation and target tracking, and instrument placement technologies; followed by system accomplishments, performance results and conclusions.

2. MISSION SCENARIO

Our mission scenario begins with the rover at the site to be explored, and assumes that a sufficiently detailed panorama of stereo images of the area has been obtained and downloaded to mission control (Figure 1). These stereo images are processed into a 3D photo-realistic virtual model of the environment, within which users explore the environment around the rover to choose target points worthy of close-up examination (Figure 2).

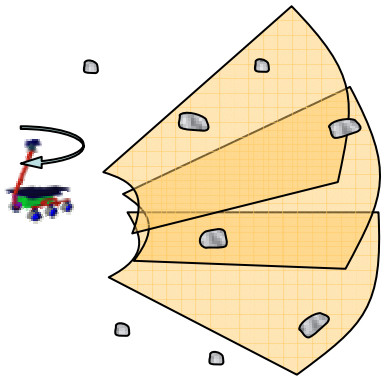

Figure 1 : *Rover has stereo image panorama of worksite at start of mission scenario.*

[2] Stephen Squyres, MER Principal Investigator, in 2004.

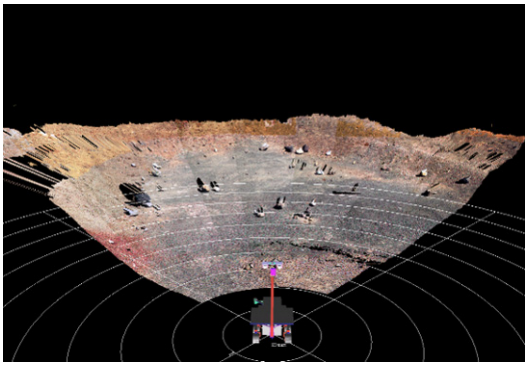

Figure 2 : *Photo-realistic 3D virtual reality model of rover environment, displayed in Viz.*

Observations are requested for each interesting target point, and prioritized with an assigned numerical value. For our purposes we assume observations are either microscopic images, requiring that the microscopic camera be placed on the target point; or science camera images taken from further away. Users must also specify where the rover needs to be in order to acquire an observation. For measurements requiring instrument placement on the target, the rover must be close enough to it that its manipulator arm can reach it. We refer to these locations as *observation points*.

In addition, users may specify additional constraints, including precedence, time of day, and whether a target should be visually tracked (required for 1cm precision microscopic camera placements).

A network of straight line path segments connecting the rover start location and all the observation points is generated. Paths avoid hazardous zones (indicated by users). Redundant paths are consolidated, and paths very likely to cause tracking failures are removed. The rover's onboard obstacle avoidance capability is sufficient to compensate for inaccurate or incomplete specification of obstacles.

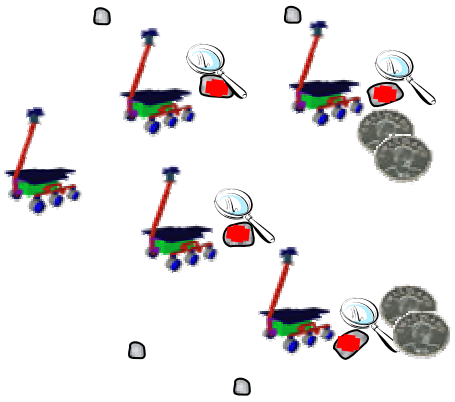

Figure 3 : *Observation requests at start of mission. For each observation, users specify a target point (red cross), what instrument (microscopic images), where the rover must be, and a subjective value for the observation (coins).*

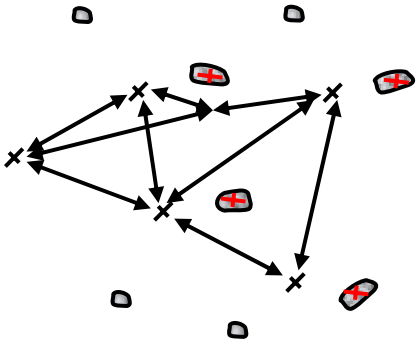

Figure 4 : *Network of paths between observation points.*

An activity plan to get the highest value measurements, subject to constraints on the total time and energy required, is generated on the ground:

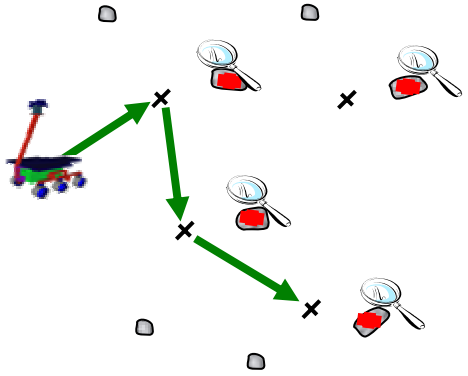

Figure 5 : *Nominal main-line activity plan to get microscopic measurements from highest value targes, subject to constraints on energy, time and target visibility.*

Points in the plan where failures are likely to occur are identified *a priori*, and contingency plans produced. Pertinent failures include losing track of a target (which will prevent accurate instrument placement on that target) or consuming excessive resources (time and energy) to get to a point, thus putting future measurements at risk.

In the example below, it can be seen that in going to the first target, the rover might lose track of a subsequent target as it leaves the region within which that target is best tracked. If this should happen, the remaining plan will break. Our solution is to insert, at the point where the failure is detectable, a contingency plan branch to acquire observations from other, still tracked, targets.

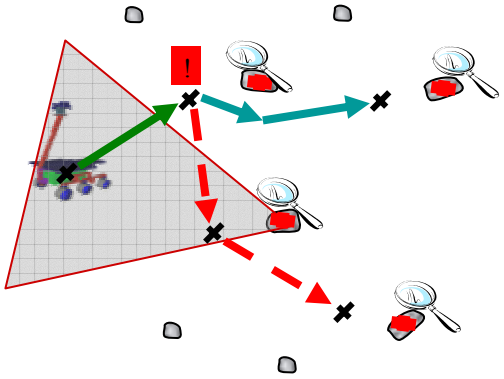

Figure 6 : *Potential target tracking failure identified when rover violates a tracking constraint, conservatively modeled here as a cone shaped region extending from the target to the rover's position when it first imaged the target. Leaving this "tracking region" implies a finite chance of losing track of the target.*

Similarly, we insert contingencies in the plan based on time or battery energy use required to get to a point. These include skipping low value targets to conserve time and energy to get to higher value targets.

Users review the plans, and make adjustments through changing constraints and observation values, adding or removing targets and editing the path networks as required. This iterative plan generation requires both a rapid planner and intuitive interfaces to quickly understand the plan generated.

The plan is converted to the *Contingent Rover Language* (CRL) used by the CRL executive running on the rover. Prior to uplink the CRL sequence is verified on a rover simulator, under the control of the same executive.

Upon receipt of the sequence, the rover commences navigating towards targets dictated by the plan, using stereo vision based obstacle avoidance and target tracking, described later. The observation locations are linked to the target frames, thus as the apparent position of a tracked target changes, so does the goal position that the rover navigates to. Once in front of targets, tracking is handed off from the main science cameras on the rover to the hazard cameras, directly over the manipulator workspace. 3D stereo models of the target area are analyzed to ensure it is safe to place an instrument against the requested target points. If not, the closest safe points on the rock are located and adjustments in the rover base position made, so that the arm may be autonomously deployed and microscopic images of the target obtained. Once done, the rover moves on to the next target until the activity sequence is completed.

Data products obtained by the rover are tagged so that they may be associated with the correct observation requests. Upon being downlinked back to mission control, users can access them through hyperlinks attached to each target in the map based UI used initially to specify targets.

3. TECHNOLOGY COMPONENT OVERVIEW

The end-to-end workflow outlined in the mission scenario description requires a robust, reliable and integrated system of components. Each component is designed optimally for a given step in the workflow but all together enable a system that aids the users – mission planners and science teams – in rapidly assessing a given mission environment, deciding what the rover execution plan should look like, interfacing to the rover during execution and relating the data products coming back from the execution back to the original context.

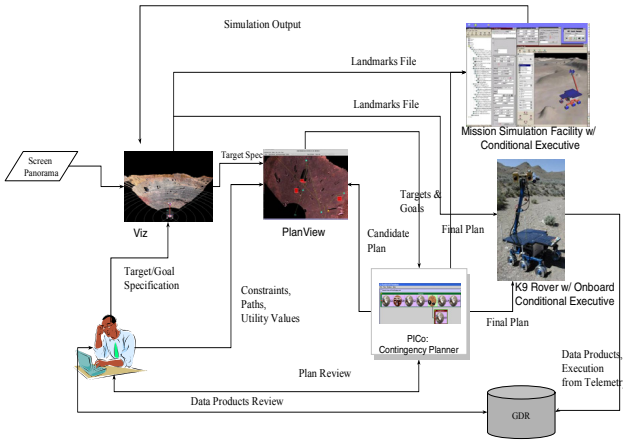

Figure 7: *System overview and information flow*

Key components of the integrated system are described next.

Viz is a 3D photo-realistic immersive display program for visualizing 3D terrain models of the area around the rover, generated from rover stereo camera panoramas. Viz was originally developed for Mars Pathfinder, and used successfully on MER for a variety of geo-morphological measurements and virtual exploration of the area surrounding the rovers 10. Our version of Viz has been significantly enhanced to allow users to rapidly specify many science targets and observations. An automatic Base Placement algorithm automatically computes the optimal Observation Point and orientation for rover that brings the target point within the arm workspace for contact measurements, where necessary. Doing this manually has proven tedious and subject to error. Besides target and observation point specification, Viz is also used to annotate the entities of interest with descriptions and specify

parameters for the observations of interest for the given execution run.

Figure 8 : *Mission Operations Center, with 3D Viz display (right) and PlanView (left).*

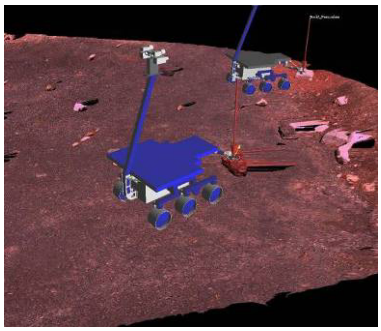

Figure 9 : *Viz, showing target selection and rover base placements in order to get microscopic measurements from the target points.*

PlanView is a large format, touch sensitive 2D user-interface, built on top of XBoard (a science collaboration tool developed for MER), for users to review requested observations and targets and associate utility values, specify additional daily mission constraints (including allowed paths, obstacles, time-of-day constraints), visualize plans and execution traces returned from the planner and rover respectively, and access returned data products. Automatic path generation, as an aid to the mission planner, allows rapid computation of reasonable traverse paths for planning purposes that takes into account a variety of constraints such as target visibility regions (computed during target specification) & obstacle information (user specified regions). These paths help specify safe traverses that abstract rover behavior for planning purposes. Actual rover execution between the points specified is determined by the onboard navigator and obstacle avoidance systems.

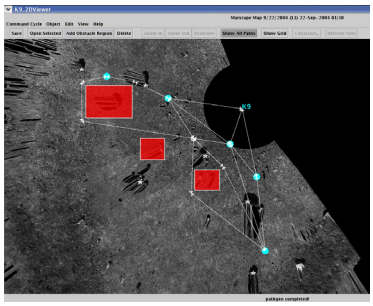

Figure 10 : *PlanView display showing observation targets, rover observation points, user defined obstacle regions and PathGen computed network of paths.*

PICo Planner is the system that develops the rover execution plans, with contingencies to deal with anticipated failures. It is driven of the observation & path specification as determined using Viz & PlanView. The generated plans are reviewed via the **PICo GUI** interface (for understanding the temporal characteristics) and the PlanView interface (for understanding the spatial characteristics) of the plan developed.

Plans developed are also validated using the **Mission Simulation Facility** [3] – a high fidelity, extensible simulation test bed configured with a simulated model of the rover and terrain of interest. Simulation validation prior to actual execution is an essential and necessary step to validate the plan, with all the built in sophistication to handle failures and contingencies, in a representative environment. Actual execution of the plan, both within the MSF and the rover, is managed by the **CRL Executive**, a robust sequence execution engine capable of handling concurrent threads of activity, temporal constraints, contingency branch selection and floating branches of contingencies. In addition to the plan sequence generated, detailed information on the target specification and tracking information for the rover is communicated via the landmarks information generated from Viz. Additional details on the PICo planner system and Contingent Executive follow in later sections in this paper.

To streamline the flow of information we adopted a central repository – the Ground Data Repository (**GDR**) – consisting of a PostgreSQL database and AFS file system. The PostgreSQL database is the primary meta-data and mission repository for information shared across the ground systems – Viz, PlanView and PICo Planner. The AFS file system is used as the platform to transfer information between the ground systems in Mission Ops Center and the rover operations in the field. The GDR allows rapid creation of mission instances that separate data and mission specification from multiple runs and also supports easy setup for what-if planning and simulation runs prior to actual execution. The GDR also serves as the primary

repository for data products coming back from rover execution.

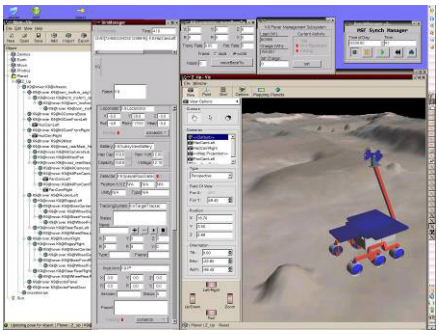

Figure 11 : *Mission Simulation Facility*

A key capability demonstrated in the integrated system is Round Trip Data Tracking – which permits intuitive and necessary tie back of the data products coming back to the original mission scenario and individual observations specified in them. This is especially critical in the given context as we could have a large number of targets and observations in a given mission scenario and an equally wide array of data products associated with each of them from execution. Managing these in a user friendly and scalable manner has been a key driver to the design and development of this integrated system.

The **K9 Rover** is a prototype planetary rover, comparable in size, capability and sensors to the MER rovers. K9 has MER equivalent drive and steering mechanisms (6-wheel steer, 6-wheel drive rocker-bogey chassis) that limits its top speed to approximately 6 cm/sec.

Figure 12 : *K9 planetary exploration rover testbed.*

K9's avionics [4] are based around a 1.2 GHz Pentium M laptop, running the Linux operating system. An auxiliary microprocessor communicates with the main CPU over a serial port and controls power switching and other I/O processing. Separate motion controllers control the wheels, camera pan-tilt and manipulator arm. Odometry sensors, a compass/inclinometer and inertial measurement unit provide information for pose estimation with error approximately 5% of distance traveled.

The K9 rover software architecture uses the Coupled Layered Architecture for Robotic Autonomy (CLARAty) [4]. Developing our technology under the CLARAty architecture decreases the difficulty of porting the system to other CLARAty robots.

Figure 13 : *K9 has front-mounted forward looking pair of b/w stereo hazard cameras overlooking the arm workspace (right); and mast mounted stereo pairs of high resolution color science cameras and wide field of view b/w navigation cameras mounted on a common pan-tilt unit such that they can acquire image panoramas from around the rover(left).*

K9's manipulator arm has 5-DOF with a total extended length of 0.79 meters. The waist yaw, shoulder pitch, elbow pitch, forearm twist and wrist pitch joints of the arm allow arbitrary x-y-z-pitch-yaw control within the arm workspace, with a positional accuracy of +/- 2 mm.

Affixed at the end of K9's arm is the CHAMP (Camera Hand-lens MicroscoPe) microscopic camera [6], a Mars Instrument Development Program (MIDP) instrument with movable CCD image plane, allowing it to obtain focused color images over a wide depth of field, from a few millimeters up to several meters, with 50 micron resolution. CHAMP has three spring-loaded mechanical distance sensors around its face that report contact with a target.

Figure 14 : *K9 manipulator arm deploying CHAMP microscopic imager on a rock target.*

4. CONTINGENCY PLANNING AND EXECUTION

PICo: Planning Incrementally for Contingencies.

Given a set of objectives and their associated values, the PICo planning system determines which of the objectives to pursue along with the detailed commands necessary to achieve those objectives. In addition, it also inserts "contingency branches" into the plan to cover situations where the plan might possibly fail. This contingency

planning is done using an incremental Just-In-Case approach [10], as illustrated in Figure 15. First a "seed" plan is generated using a conventional planning system, assuming that actions have their expected outcomes. This plan is then evaluated to determine where it might fail, given information about the probability of failure for the different actions, and information about the uncertainty in time and resource consumption of the actions. A branch point is then chosen using heuristics that estimate where a branch is likely to most improve the overall expected utility of the plan. An alternative or *contingency* plan is then constructed for this branch, and incorporated into the primary plan. The resulting conditional plan is again evaluated, and additional branches can be added as needed, either to the original mainline plan, or to already existing contingency branches.

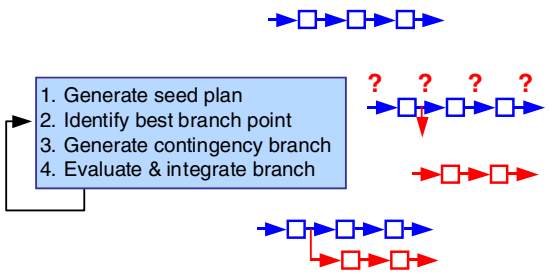

Figure 15 : *PICo algorithm*

The software architecture for PICo is shown in Figure 16. The contingency planner makes use of the PLASMA planning engine to generate seed plans, and to generate the plans for the contingency branches. PLASMA is a constraint-based temporal planning engine [11][16] developed at ARC as the successor to the EUROPA planning system [11] currently in use as part of the MAPGEN software that generates daily command sequences for the two MER rovers, Spirit and Opportunity. To construct a seed plan, the contingency planner gives PLASMA a subset of the possible goals, expected resource availability, and expected resource consumption of actions. When the plan comes back, the contingency planner evaluates it using a Monte Carlo simulation to determine the impact of possible tracking failures, and uncertainty in both time and resource usage. To build the branch, the planner again passes an appropriate subset of the goals, the state of the rover at the branch point, and resource availability to PLASMA. The state of the rover and the resource availability is based on the branch condition and includes the amount of resources (time and energy) available, and the tracking status of the different targets.

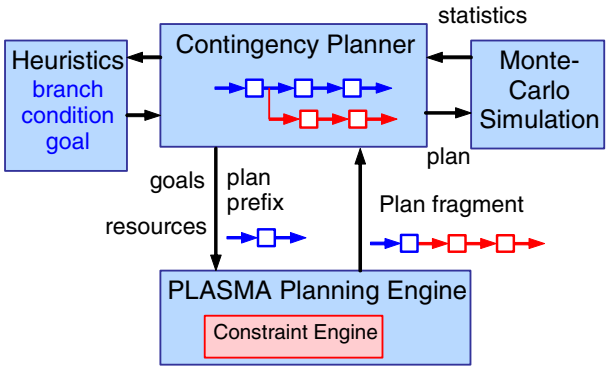

Figure 16 : *Architecture of the contingency planner*

The problem of automatically choosing good branch points and good branch conditions is quite hard in general [12]. Intuitively, it might seem that a good place to put a contingency branch is the place where the plan is most likely to fail. While this works reasonably well for discrete failures like tracking failure, it does not work well for failures associated with consumption of continuous quantities like time and resources. The problem is that time and resources are typically exhausted near the end of the plan. With few resources remaining, there may not be any useful alternative plans. Instead, one would like to anticipate impending failures earlier in the plan, when useful alternatives remain. In other words, the planner wants to find point(s) in the plan where a contingency branch would significantly increase the overall utility of the plan. Of course, this is difficult to determine without examining and constructing plans for every possible contingency branch, which would be computationally intractable. Instead, we use heuristics. Among the many different heuristics we have implemented and evaluated, the most effective is to select points where there is high variance in the utility of the plan. For example, suppose there is considerable uncertainty in the amount of time or energy consumed by a particular drive operation. This could impact whether or not science measurements at the end of the plan are possible. As a result, there is considerable variance in the utility of the plan following this drive action. To compute this variance for a given point in the plan, we need to know two things:

- the utility of the remainder of the plan as a function of the resources available
- the probability distributions for resource availability at that point

Both of these distributions are estimated using Monte Carlo Simulation. Given these distributions, it is straightforward to compute the variance in utility that one expects at this point in the plan.

Given a particular branch point, we must also select the branch condition. For tracking failure, this is trivial; we branch if tracking has failed. However, for continuous resources it is more difficult. In general, the decision depends on and influences the plan that we construct for the contingency branch. If we have a branch plan with high utility, we could be more aggressive about branching, but if the branch plan has low utility, we would like to be more conservative. Unfortunately, we need to first choose a condition in order to generate a good branch plan. Again, we resort to heuristics for doing this. A simple but effective heuristic is to branch when the resources remaining are less than 80% of the expected consumption for the remainder of the plan. A somewhat better heuristic would be to branch when the remaining resources are such that the expected utility of the remainder of the plan drops below some threshold.

In addition to branch and condition selection, the contingency planner must also decide which goals to pursue, both for the original seed plan, and for each contingency branch. This is because the fundamental problem is an *over-subscription* problem – there are more objectives than possible, given the time and resources available. Few planning systems are designed to deal with over-subscription, and those that do, typically adopt a greedy approach to selecting goals. For rover problems, we have found that goal selection is critical to obtaining high quality plans. As a result, we use a sophisticated method for selecting goals, based on the solution of a deterministic *orienteering* problem [14]. An orienteering problem is a variant of a traveling salesman problem in which there are rewards in various cities, the salesman has a finite amount of gas, and the objective is to collect as much reward as possible before running out of gas. Figure 17 shows a rover problem recast as a deterministic orienteering problem: the cities become target sites, and the roads are paths between different targets, with costs corresponding to the resources required for the rover to traverse the path. The prizes are the scientific values of the experiments at a given target site. However, since there can be multiple experiments possible at a given site, and there are time and resource costs associated with each experiment, we create a separate "city" in the graph for each experiment at a site. We then add directed edges from the site to the experiments at that site, with costs corresponding to an estimate of the resources required to do that experiment.[3]

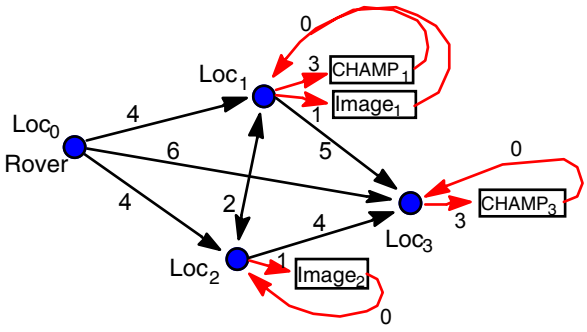

Figure 17 : *Orienteering graph for a rover problem.*

The solution of this orienteering problem results in an ordered set of goals that can likely be achieved given those resources. This information is then used by the contingency planner to determine appropriate goals for the PLASMA planning engine.[4]

To summarize, the contingency planner first constructs and solves an orienteering problem to determine an appropriate subset of the goals. This goal set is then fed to the deterministic PLASMA planning engine to produce a seed plan. The seed plan is evaluated using Monte-Carlo Simulation, and the resulting statistics are used by the branch selection and branch condition heuristics to propose an appropriate branch. A new orienteering problem is constructed and solved to determine an appropriate goal set for the branch, and PLASMA is invoked again to produce the branch plan. The resulting branch plan is integrated into the mainline plan, evaluated using Monte-Carlo, and the process continues.

Once planning is complete, the resulting contingency plan is passed back to PlanView for display, and to a GUI shown in Figure 18.

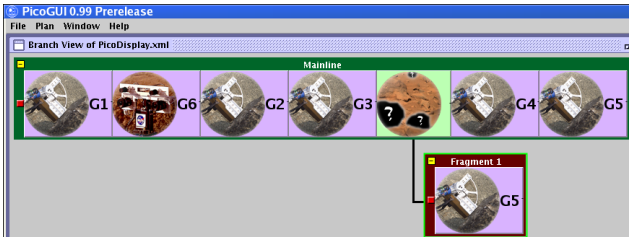

Figure 18 : *GUI display of a simple contingency plan with one branch on tracking failure of target G4. Icons indicate experiment types and branch types.*

[3] Experiment costs can be estimated using plangraph cost estimations techniques. See [15] for details.

[4] The orienteering graph is only an abstraction of the real planning problem, since it leaves out time constraints and other details involved in performing the science experiments. As a result, the solution is only an approximation and may not be optimal or even possible.

Once the plan is deemed satisfactory, it is shipped to the rover onboard executive.

Robust Execution

The CRL Executive is responsible for interpretation of the contingency plan coming from the ground and generated by the contingency planner. The CRL Executive is designed to be more capable than traditional sequence execution engines; it can handle the expressive plans generated by the contingency planner and can perform limited plan adaptation itself. The current features being supported by the Executive include:

- Concurrent threads of activity
- Temporal constraints
- Contingency branches
- Floating branch insertion.

The planner translates its plan into the Contingent Rover Language (CRL) for uplink, and the CRL Executive interprets the CRL-encoded plan directly. CRL is a flexible, conditional sequence language that allows for execution uncertainty [8]. CRL expresses temporal constraints and state constraints on the plan, but allows flexibility in the precise time that actions must be executed. Here is an excerpt of a CRL converted plan, that includes starting the tracking of rocks, and a branch whose plan controls the rover to navigate to a rock and perform an arm placement:

```
(task :id t34992 :probability 0
      :continue-on-failure nil
      :start-temporal-conditions
        ((t34930 end (0 :plus-infinity))
         (2 268435456))
      :end-temporal-conditions ((3 268435456))
   :action trackstart
      :parameters (name = "Rock1")
      :expectations ((time 1 0)(energy -1 0)))
(branch :id branch3 :options
  ((option :id option4 :utility 1.0
           :eligible-conditions
             ((database (:trackerstate … )))
           :node
      (concurrent-block :id block5
        :continue-on-failure nil
        :node-list
          ((block :id t73453
             :continue-on-failure nil
             :start-temporal-conditions …
             :end-temporal-conditions …
            (task :id t73297 :probability 0.9
                  :continue-on-failure nil
                  :start-temporal-conditions …
                  :end-temporal-conditions …
                  :action navigate
                  :parameters
                   (wrt_frame="W1" …  )
                  :expectations
                    ((time 1457 1280)
                     (energy -174345 -49250)))
           (task :id t73775 :probability 0
             :continue-on-failure nil
                 :start-temporal-conditions …
                 :end-temporal-conditions …
              :action ipplaceinstrument
          :parameters (name="current")
    :expectations
                 ((time 504 120)
                  (energy -34611 -85)))
```

The structure of the CRL plan language and its interpretation are completely domain-independent. Domain-dependent information is added by specifying a command dictionary, with command names and argument types, and a command interface, which passes commands to the rover and return values and state information from the rover.

The CRL Executive is responsible for interpreting the CRL command plan coming from ground control, checking run-time resource requirements and availability, monitoring plan execution, and potentially selecting alternative plan branches if the situation changes. At each branch point in the plan, there may be multiple eligible options; the option with the highest expected utility is chosen. For this demonstration, the contingency planner generated mutually exclusive branches.

The Executive maintains a queue of events and expands them one after the other, or concurrently, according to the current context. Figure 19 shows the expansion of a single task. Expansion of a (concurrent) block of tasks is performed similarly.

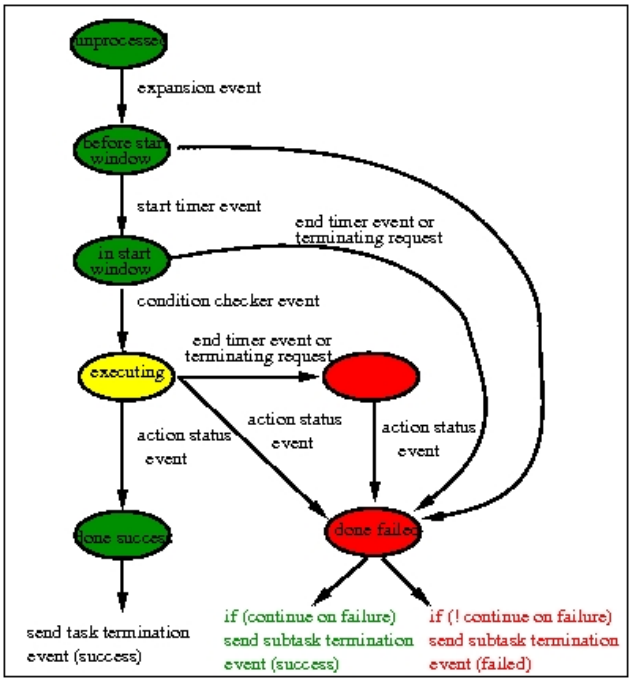

Figure 19 : *Execution of a CRL task.*

Temporal constraints allow interactions among actions to be limited, for example to specify that an instrument should be used shortly after it is warmed up, or to specify that image taking should occur after driving. In addition, absolute temporal constraints are needed to specify appropriate times of day to perform actions, such as when illumination is sufficient for the spectrometer to operate. The Executive supports constraints expressed as temporal intervals that delimit the possible times that a task may start or finish with respect to other tasks, or in absolute time.

A primary feature of CRL is its support for contingency branches to handle potential problem points or opportunities in execution. The contingency branches and the flexible plan conditions allow a single plan to encode a large family of possible behaviors, thus providing responses to a wide range of situations. The Executive maintains a database of system state values (e.g. current battery level, status of tracked targets, elapsed time since plan execution started) against which branch conditions are repeatedly tested.

A novel feature of the CRL Executive is its support for "floating contingencies," which are plan fragments that may be inserted at any point in execution [8]. For example, a plan to perform opportunistic science during a traverse is naturally expressed as a floating contingency, since the presence and position of an interesting science target is unknown before the traverse. Likewise, a plan to stop and recharge the battery is another example of a floating contingency. In general, floating contingencies would be impractical for the planner to consider because of the large number of possible branch points that they would add to a plan, enlarged continuous space, the use of a nominal model of the world and actions that do not represent faults that can affect the system. Therefore plans do not take into account numerous situations. Existing modules enhanced the current Executive with more capability to react to non predictable events, through the re-evaluation of plan branches value given an estimate of the rover state, as returned by an onboard diagnosis engine.

The CRL Executive is implemented as a multi-threaded, event-based system (see Figure 20). Around a central Executive event-processing loop are threads to handle timing, event monitoring, action execution monitoring, and telemetry gathering. The central event processor sends requests to the other threads (for example, "wake up at time 20" or "notify when battery state of charge is below 4Ah") and receives events relevant to those requests. This architecture allows the CRL Executive to support concurrent activities and flexible action conditions expressible within the CRL language.

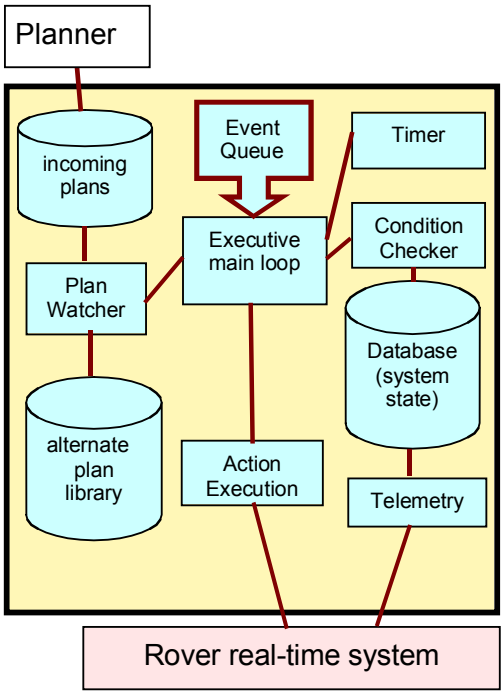

Figure 20 : *CRL Executive structure. The main event loop communicates with other threads for services such as timing, action monitoring, and event monitoring. External connections are to a planner, which supplies new plans to execute, and a rover real-time system, which executes actions and supplies telemetry data.*

5. NAVIGATION

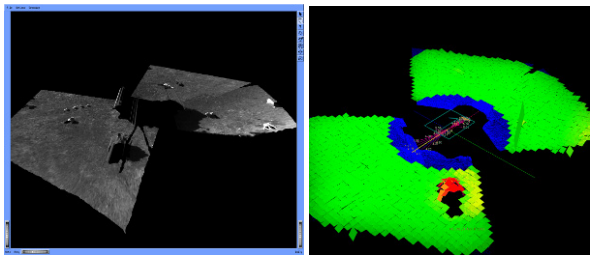

Figure 21 : *The CLARAty Navigator stereo model and traversibility map.*

Local path following and obstacle avoidance is provided by a Navigator which is bundled in the CLARAty[4] robot software architecture. The Navigator is the next iteration in a lineage of robot navigation implementations including the Morphin[20] algorithm used by the Nomad rover at CMU, and the GESTALT navigator on the MER rovers.

The rover moves in short steps, stopping momentarily between each to analyze the terrain. At each step, the navigator takes four pairs of images from the rover's navigation cameras, at set pan/tilt angles. These images cover the area that the rover may choose to move through.

The images are converted into 3-D point clouds using stereo, and the point clouds are quantized into a grid, where each grid cell contains a local plane fit of the points that lie within it. The traversability of the grid is determined by the residual of the plane fit, the roll and pitch angles of the plane, and the height difference between the cell and adjacent cells. A confidence measure is estimated from the number of points that fall inside the cell. Each cell is given a cost, which combines local obstacle information with global information about the goal location and other obstacle regions in the cumulative map. Arcs through the grid are then analyzed, and the arc that accumulates the least cost as it passes through each cell is chosen for the rover to move along. At each step, a new map is added to the cumulative map, which is discounted to allow for drift.

6. VISUAL TRACKING

We have developed a combined feature based and shape based visual tracking system that leverages the benefits of each method in a complementary manner.

In order to handle large errors in platform motion prediction, reduce tracking frequency, and handle targets which do not facilitate unambiguous appearance based matching, the system includes a tracker which makes use of invariant feature detection and matching. The SIFT feature detector [17] finds large populations of features around the target of interest. By matching features across a stereo pair, as well as matching pairs before and after robot motion, the tracker can quickly compute a 6-DOF motion, and in a static environment this 6-DOF transformation describes the motion of the tracked point. RANSAC [18] is used to provide robustness to errors during feature detection and matching. RANSAC finds the largest set of putative matches that can be aligned with a single rigid body transformation, rejecting as outliers those points that cannot be aligned.

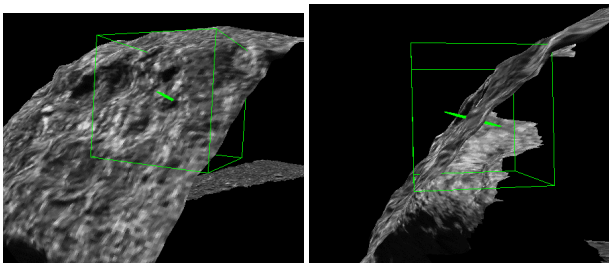

Figure 22 : *Initial target location uncertainty due to subpixel errors in target selection and stereo matching.*

In addition to tracking the nominal location of the target, the SIFT based tracker also estimates the uncertainty in the tracker estimate. The initial target selection uncertainty is found using the unscented transform [19], assuming a half-pixel error in the specification of the pixel coordinates in the initial science camera view. This unscented transform on the subpixel location of the target in the left and right stereo cameras yields a covariance matrix over the XYZ position of the specified target, which is shown in Figure 22 as an ellipsoid rendered with a stereo model of the rock of interest. Bootstrap is used to recover the uncertainty in each incremental update by recovering the optimal transformation under many random subsets of the inliers found with RANSAC, and the uncertainty in the target location is compounded with the uncertainty in the transform to estimate the uncertainty in the target location after each tracker update. This uncertainty region can also be rendered into the rover camera viewpoint for visualization purposes (Figure 23).

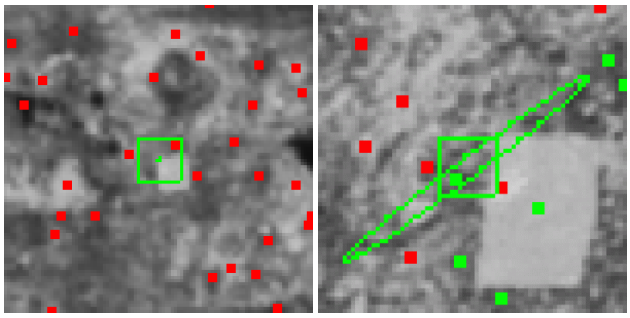

Figure 23 : *Initial target specification from 10 meters away. The upper left corner of the rectangular marker is chosen as the target of interest. In the final image after robot navigation, the tracked point is only a few pixels away and the correct point is well within the covariance ellipse.*

In addition to reasoning about the uncertainty in the target location, represented by the target location covariance matrix, the tracker also maintains a measure of confidence that the target is still being tracked properly. This confidence measure is a function of the number of points that match between successive views. If RANSAC finds a large number of inliers, then the reported confidence is high. If RANSAC cannot find a solution or only finds a small number of inliers, then the confidence drops. Several low confidence updates in a row will lead to very low confidence and below a threshold the tracker will simply report failure. Tracking failures are then handled at the executive level by switching to a contingency branch for the failed target tracking, and the rover might for example abort a target approach and instead visit another target. Figure 24 shows several plots of tracker confidence vs. distance traveled during two different multi-target instrument placement tests.

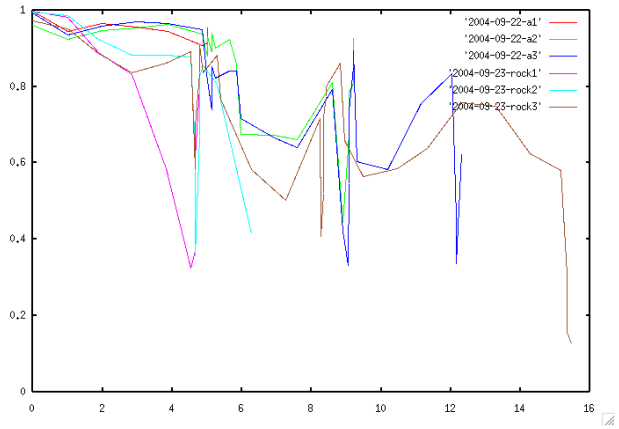

Figure 24 : *Tracker Confidence vs. distance for 6 different tracked targets during 2 different multi-target instrument placement tests. This confidence measure is used to decide when to abort a target approach due to tracking failures.*

If RANSAC cannot find any matches in two consecutive views, it will not immediately report a failure. The last set of valid interest points are retained and matched against incoming images until either a match is found or enough updates have occurred without success for the tracker to abort tracking. Typically the tracker will try 4 or 5 times before the confidence falls below the threshold, and in a few tests the tracker has actually recovered a lost target.

Because the motion recovered by the feature based tracker is incremental, compounding the transformations leads to target drift over time. This is compensated by making use of a shape based tracker. The shape based tracker employs 3D terrain model registration based on nonlinear optimization. Model registration can provide a strong cue for recovering 6-DOF motion if the models have sufficient shape and the optimization is initialized sufficiently close to the solution. By using the output of the feature based tracker to initialize the registration, we are able to align the current target view to the original view, thereby eliminating drift incurred by the feature tracker's incremental recovered motion.

The 3D registration method is based on matching range images under different hypotheses for the transformation between views until the transformation that best aligns the models is found. Given two meshes generated from two different views, the shape based tracker makes use of a virtual range sensor in order to determine depth at each corresponding point between the two meshes. By minimizing the difference between the rendered depths at each point, we can extract a rigid transformation that aligns the two models, thereby allowing us to determine the coordinate transformation between views. The rendering step is fast and eliminates solving a separate correspondence problem using nearest neighbor heuristics such as ICP. Figure 25 and Figure 26 show results from the 3D registration.

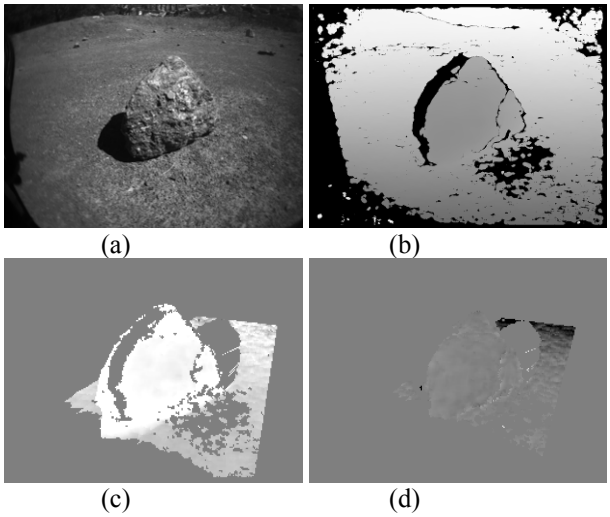

Figure 25 : *Registration result. (a) Hazcam images taken when rover arrives at target. (b) Depth map from stereo. (c) Minimum depth error from correlation search. (d) Depth error after Nelder-Mead optimization.*

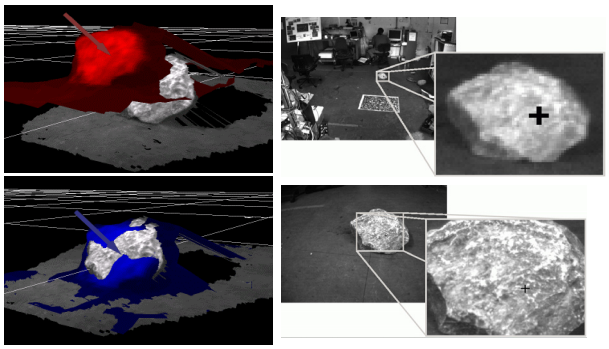

Figure 26 : *The combined tracking system is capable of tracking user specified points for robotic navigation with centimeter accuracy over tens of meters.*

7. INSTRUMENT PLACEMENT

Once K9 arrives at a goal location, the target location must be found in the rover mounted hazard avoidance cameras. These cameras provide the best view for arm motion planning. The target handoff uses the mesh registration algorithm described previously.

Once the goal location is known in the hazcam reference frame, the target is segmented from the background and assessed to find safe places for instrument placement. The segmentation extends the EM algorithm described in [22] to cluster the 3D points in a point cloud from the stereo cameras into a dominant ground plane and an a priori unknown number of clusters representing rocks in the foreground. The algorithm begins by finding the ground by fitting a plane to the entire point cloud using RANSAC,

which results in the plane containing the largest number of points from the point cloud. Connected regions of points which do not match the ground plane are initially labeled as rocks, and then an EM algorithm alternates between labeling points in the point cloud as either ground or rocks, then fitting the parameters describing the ground plane and rocks given the labeling. A typical result is shown in below.

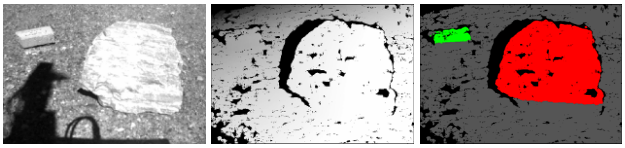

Figure 27 : *Hazcam image, depth image, and segmentation result.*

Once the rock has been segmented from the ground, it is searched for places that the instrument can be safely placed. The priority is for the target point selected by the user, but if that location is unsafe the rover will find an alternate placement location. The target safety check uses a few heuristics. The first is the tool radius, which is the radius of the circle circumscribing the contact interface. For any point on the rock, all points within this radius are used in a planar fit. If there are any points that deviate from that plane by too much, the placement location is marked unsafe. This is to prevent protrusions from scratching the microscope lens. The next heuristic is the validity of stereo in the neighborhood. Since some pixels in the reference camera do not have a valid match in the right camera view, the stereo model is incomplete. A certain percentage or certain maximum hole size may be tolerable, but if there is too little valid stereo or too large a hole in the model, the point is labeled unsafe. Finally, if the angle between the viewing direction and the surface normal (tool orientation) is too great, the point is labeled unsafe because the stereo cameras may not have a good enough view of the surface from the orientation of the tool placement.

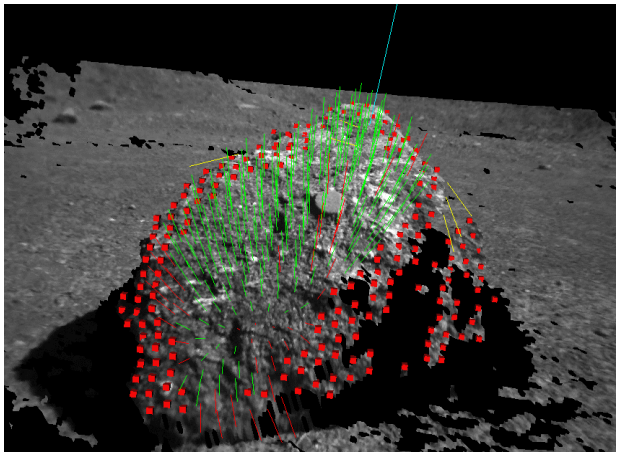

Figure 28 : *Instrument safety check result. Green lines indicate safe placement locations. Blue line indicates point selected for instrument placement. Red cubes indicate points that were discarded due to lack of good stereo.*

8. PERFORMANCE & ACCOMPLISHMENTS

On October 28, 2004 we conducted a live integrated demonstration of this system and mission scenario before a distinguished audience of Mars Science Laboratory (MSL) mission managers and engineers, a representative of the MER science team and various other scientists and program managers from NASA and academia.

The audience spent the morning in the Mission Ops room, reviewing a 3D panorama of the rover worksite (Figure 29), choosing targets and observations, and generating an activity plan with contingencies in case of insufficient time or inability to track targets. One of the targets was chosen by the MER science team member. The generated activity plan was verified in simulation and uplinked to K9 in the dry lakebed area of the Marscape. During the afternoon, in less than 3 hours, the K9 rover traversed 28m, successfully tracking and approaching four science targets to acquire close up microscopic images of three. Instrument placement on the fourth target was autonomously rejected by the rover on the grounds that insufficient stereo data was available for safety to be guaranteed. Interestingly, the farthest identified target (at 10 meters distance) was not approached. The planner correctly determined the target was not as valuable as other closer alternatives.

Figure 29 : *NASA Ames Marscape planetary rover test facility. Marscape is ¾ acres in size, with a diverse and realistic Mars analog topography, including a dry streambed draining into a dry lakebed that overlaps an old meteorite impact crater and other presumed Martian geological features. A nearby operations trailer provides wireless networking coverage.*

Accuracy and Distance

We conducted tests on 9/22/2004 – 9/23/2004 at the Marscape test site to gauge the accuracy, reliability and

distance limitations of the target tracking, navigation and instrument placement systems (Figure 30).

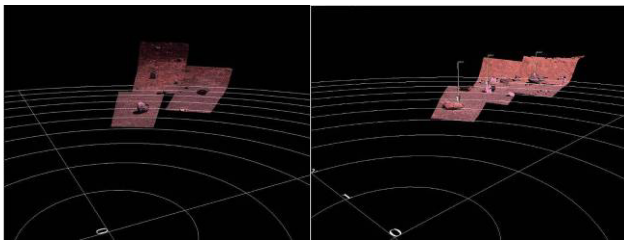

Figure 30 : *Test target arrangements on 9/22/2004 and 9/23/2004. Rover started at origin and was commanded to navigate sequentially to each rock target and place the microscopic imager at operator designated target points (artificially marked). The target rocks are approximately 5m, 7.5m and 10m distant from the origin.*

Tables 1 and 2 show the results for these two days of testing, along with tracker images of each of the targets.

For each target, we recorded the elapsed time of the traverse (which can be large if the rover must drive around several obstacles), the accuracy of the target as tracked by the feature-based tracker relative to 3-D models generated by the same camera pair as is used in the tracking, and the accuracy of the 3-D shape-based tracker used for hand-off from the science cameras to the hazard cameras (for final placement). The distance of the final (safe) placement point from the designated point is also recorded. This distance can be arbitrarily large since the system places a higher priority on safety than on accuracy of placement. Placement figures are not available for September 22nd; a motor failure in one of the arm joints prevented successful placement.

The only failure in tracking occurred in the feature-based tracker for the second rock on September 23rd when the rover cast a large shadow into the scene. Once the tracker was unable to find a transformation between subsequent images, it stopped updating the target location and reverted to dead reckoning. After the navigation was finished, the shape-based tracker was able to recover the target with accuracy comparable to the other experiments.

Table 1 *Performance Characteristics from Test Run 1 (9/22/2004)*

Target	1 (5m)	2 (7.5m)	3 (10m)
Time to reach target	21 mins	+42 mins	+17 mins
Tracker accuracy	0.68 cm	0.29 cm	1.3 cm
Hand-off accuracy	0.5 cm	2.7 cm	1.7cm
placement distance	n/a	n/a	n/a
Target 1	Target 2	Target 3	

Table 2: *Performance Characteristics from Test Run 2 (9/23/2004)*

Target	1 (5m)	2 (7.5m)	3 (10m)
Time to reach target	25 mins	+27 mins	+23 mins
Tracker accuracy	~ 0.3 cm	failed	1.7 cm
Hand-off accuracy	1.51 cm	~1.6cm	2.7 cm
placement distance	~6.3 cm	~11 cm	~ 3 cm

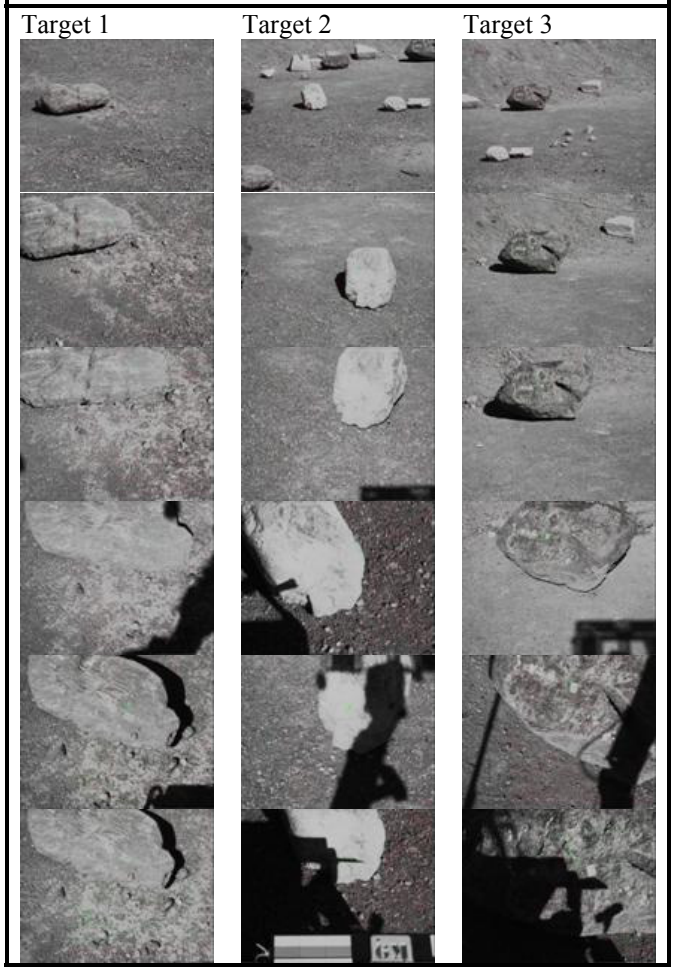

Figure 31: Keypoint tracker images

Usability

Activity	Time required

Preliminary Site Analysis	As required: Typically 1 – 2 hours.
Target Requests	~3-5 minutes/target
Observation Requests	~ 1-3 minutes/target
Path Specification	10 sec per obstacle 10 sec to generate paths ~ 2-5 minutes overall
Activity Planning	~ up to 10 minutes overall, depending on plan quality. Planner takes up to 3 minutes to generate a new plan.

Overall System Performance

Up to and including the aforementioned demonstration, our system has achieved the following:

(1) Instrument placement on 4 targets (1:23 hrs execution).

(2) Targets up to 10m distant.

(3) Total traverse distances > 28m whilst tracking targets and executing plan.

(4) Up to 1cm instrument placement accuracy.

(5) Tracking failure avoidance and recovery.

(6) Time and resource monitoring and recovery successfully anticipated and avoided future failures.

(7) Round trip data tracking.

(8) 5 person operations staff.

Qualitatively, once we had the plan running on the rover, we never ran into a situation in which the exec failed because of an unplanned contingency, nor did instrument placement fail once we reached a rock (though our system did successfully decide on occasion to not attempt instrument placement if no safe locations on the rock were found).

The tracker performed sufficiently well to recover targets that had been temporarily lost, and we had to write code to inject faults into the system in order to test the plan's contingency branches. In every case where the feature based tracker failed after the plan branch point, the shape based tracker was able to recover the target once the rover arrived.

9. CONCLUSIONS

We have successfully demonstrated a complete integrated system for multi-target single cycle instrument placement, meeting or exceeding early MSL requirements and representing a tenfold increase in capability over the current flown state-of-the-art (MER).

With current technologies, the MER rovers currently operating on Mars would require up to 12 sols to accomplish what our system does in 1 day. Further, the instrument placement demo was accomplished with a control staff of 5 persons, as contrasted with the hundreds of ground-based operators currently required for MER.

Whilst considerable work remains to exhaustively test and validate our component technologies for potential mission infusion, there are other factors that may delay their uptake into missions like MSL.

The current MSL mission scenario calls for intensive, long duration analyses of each rock it encounters, thus eroding the relative value of getting to them quickly. Nevertheless, even with a presumed 5 sol dwell time per rock, this single cycle placement capability implies a 30% increase in productivity. Furthermore, it is scientifically desirable to survey a site to find the interesting targets before investing significant resources to analyze them. Such an initial survey, using contact or close up sensors, is well served by this system. In particular, our multi-target tracking capability enables us to easily return to previously investigated targets that the science team later deem interesting.

Looking ahead to the future, our vision is to build truly capable robotic field assistants that can rapidly and comprehensively survey a site with both images and contact measurements, enabling human like work practices with 10x to 100x increases in science return, commands being executed in minutes, not days; and smaller operations teams.

A recent study of the hyper-arid regions of the Atacama Desert indicates that only 0.08% to 0.1% of rocks there harbor microbial colonies. To the degree that this is indicative of the difficulties of finding evidence for extinct or extant life on Mars, it is clear that thousands of rocks there would need to be examined. This work is a step towards realizing the vision that would make this class of investigation possible.

10. REFERENCES

[1] Krasner, S.M., Tamppari, L., Steve Peters, S., Limonadi, D. (2002), "MSL Scenarios and Autonomy Requirements", MPSET meeting 4/11/2002.

[2] Stoker, C., E. Zbinden, T. Blackmon, B. Kanefsky, J. Hagen, C. Neveu, D. Rasmussen, K. Schwehr, M. Sims (1999), "Analyzing Pathfinder Data Using Virtual Reality and Super-resolved Imaging," Journal of Geophysical Research, vol 104, no E4, pp. 8889-8906, April 25, 1999.

[3] Flückiger, L. and C. Neukom, (2002), "A new simulation framework for autonomy in robotic missions," *in* proc. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, September 30 – October 4, 2002

[4] Eric Park, Linda Kobayashi, and Susan Y. Lee. Extensible hardware architecture for mobile robots. In *IEEE* International *Conference on Robotics and Automation*, 2005, under submission.

[5] R. Volpe and et al. The claraty architecture for robotic autonomy. In *Proceedings of the 2001 IEEE Aerospace Conference*, 2001.

[6] G.M. Lawrence, J.E. Boynton, and et al. Champ: Camera handlens microscope. In *The 2nd MIDP Conference, Mars Instrument Development Program*, 2000.

[7] J. Bresina, R. Dearden, N. Meuleau, S. Ramakrishnan, D. Smith and R. Washington, "Planning under continuous time and resource uncertainty: a challenge for AI," *19th Conference on Uncertainty in AI*, 2002.

[8] J. Bresina and R. Washington. "Robustness via run-time adaptation of contingent plans," *AAAI Spring Symposium: Robust Autonomy,* 2001.

[9] R. Dearden, N. Meuleau, S. Ramakrishnan, D. Smith and R. Washington, "Incremental contingency planning," *ICAPS-03 Workshop on Planning under Uncertainty,* 2003.

[10] M. Drummond, J. Bresina and K. Swanson, "Just-In-Case Scheduling," *12th National Conf. on Artificial Intelligence (AAAI-94),* 1994.

[11] J. Frank and A. Jónsson, "Constraint-based attribute and interval planning," Constraints, 2003.

[12] N. Meuleau and D. Smith, "Optimal limited contingency planning," *19th Conf. on Uncertainty in AI,* 2003.

[13] L. Pedersen, M. Bualat, D. Smith and R. Washington, "Integrated Demonstration of Instrument Placement, Robust Execution and Contingent Planning," *7th International Symposium on Artificial Intelligence and Robotics in Space (iSAIRAS-03)*, 2003.

[14] C. Keller, "Algorithms to solve the orienteering problem: a comparison," *European J. Operational Research* 41, 224–231, 1989.

[15] D. Smith, "Choosing objectives in over-subscription planning," *14th Intl. Conf. on Automated Planning and Scheduling (ICAPS-04),* 2004.

[16] D. Smith, J. Frank, and A. Jónsson, "Bridging the gap between planning and scheduling," *Knowledge Engineering Review* 15(1), 2000

[17] D. G. Lowe. Distinctive image features from scale invariant keypoints. *International Journal of ComputerVision*, 60(2):91.110, 2004.

[18] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model _tting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381.395, June 1981.

[19] S. Julier and J. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, 1997.

[20] C. Urmson, R. Simmons, and I. Nesnas. A generic framework for robotic navigation. In *Proceedings of the 2003 IEEE Aerospace Conference*, 2003.

[21] Deans, M., C. Kunz, R. Sargent, L. Pedersen, "Terrain Model Registration for Single Cycle Instrument Placement," in *International Conference on Intelligent Robots and Systems (IROS),* Las Vegas, USA, 2003.

[22] Pedersen, L., (2002) "Science Target Assessment for Mars Rover Instrument Deployment," in *proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, September 30 – October 4, 2002.

11. BIOGRAPHY

Liam Pedersen is Principal Investigator for the Multi-Target Single Cycle Instrument Placement (IS Level 1 Milestone) project described in this paper. He did his Ph.D. work at the Carnegie Mellon University Robotics Institute prior to joining the Intelligent Robotics Group at the Computational Sciences Division of NASA's Ames Research Center. His prior work includes a Bayes network based rock classifier used by a robot to autonomously identify meteorites in Antarctica; automated instrument placement for planetary rovers and various robotic field campaigns in polar regions and deserts across the world. In addition to this work, he leads a project to build a solar and wind powered rover for a biological survey of Antarctic ice sheets.

Srikanth Rajagopalan is Project Manager for the Multi-Target Single Cycle Instrument Placement (IS Level 1 Milestone) project at the Computational Sciences Division, NASA Ames Research Center. He has 10 years of industry experience managing and coordinating large scale, distributed software projects and application development. Prior to his position as Project Manager at NASA Ames Research Center, Srikanth was Director for Software Development at i2, a leading supply chain solution provider. Besides software program management he has demonstrated background and experience in Planning & Scheduling applications. Srikanth has an MS in Industrial Engineering from Purdue University.

David E. Smith is lead of the Planning and Scheduling Group at NASA Ames Research Center. He has been involved in the development of several influential planning systems, including CNLP, CGP, SGP, TGP, and Fragplan, which have pushed the envelope of automated planning technology in the areas of uncertainty, continuous time, and over-subscription. Dr. Smith has served as an Associate Editor for the *Journal of Artificial Intelligence Research*, and editor for special issues on the Third and Fourth International Planning Competitions. He received a Ph.D. in Artificial Intelligence from Stanford University in 1985.

Matthew Deans did his PhD work at the Carnegie Mellon University Robotics Institute, before joining the Intelligent Robotics Group at NASA Ames Research Center. He has developed simultaneous localization and mapping (SLAM) algorithms and sensor fusion based localization systems for rovers deployed in desert field sites in California, Nevada, Chile and Antarctica. He has also developed and supported machine vision ground tools used in MER mission science operations.

David Lees is a researcher in the Intelligent Robotics Group at NASA Ames Research Center where he studies 3D visualization techniques and surface reconstruction methods (most recently applied to the Mars Exploration Rover mission), as well as novel user interfaces for command and control of planetary rovers.

Clayton Kunz is the lead software engineer for the K9 rover at NASA Ames, and spends much of his time working on computer vision problems for robotics. He is also the head of the math and data structures subgroup of CLARAty, a collaborative project developing a software architecture for robotic autonomy. He is an employee of QSS Group, and has had his hands inside K9, at Ames since 2001, before which he spent time making robot tour guides at a start-up company in Pittsburgh, PA. Clay holds BS and MS degrees from Stanford University, and lives in San Francisco.

Randy Sargent is software lead for the K9 rover target approach and instrument placement system at NASA Ames since 2002. In 1994 Randy co-founded Newton Research Labs, a machine vision company, with Anne Wright and Carl Witty. Randy led the Newton Labs team which won the 1996 and 1997 MIROSOT robot soccer tournaments, as well as the 1996 AAAI "Clean up the Tennis Court" robot contest. Randy left Newton Labs in 2000 to join Blastoff!, and in 2001 became Director of Open-Source Robotics at the KISS Institute for Practical Robotics. Randy holds BS and MS degrees from the Massachusetts Institute of Technology.

12. ACKNOWLEDGEMENTS AND PERSONNEL

This project was made possible with both funding from the Intelligent Systems (IS) Program, with additional contributions and support from the Astrobiology Science and Technology for Exploring Planets (ASTEP) and Mars Technology programs.

We wish to acknowledge the continued and strong support of James Crawford, Daniel Clancy, Butler Hine and Robert A. Morris at NASA Ames Research Center, without which this project would not have been possible

A large team of individuals has been involved in the development and demonstration of this technology. We list them below.

PI & PM: Liam Pedersen, Srikanth Rajagopalan

Navigation, Instrument Placement and K9 Rover: Randy Sargent, Matt Deans, Maria Bualat, Clay Kunz, Anne Wright, Eric Park, Susan Lee, Linda Kobayashi, Hoang Vu, Sal Desiano, Alan Chen, Matt Mclellan, Ted Morse.

Contingency Planning: David Smith, Nicolas Meauleau, Sailesh Ramakrishnan, David Roland, Matthew Boyce, Betty Lu.

CRL Executive: Richard Washington, Howard Cannon, Ray Garcia, Emmanuel Benazera.

Ground Data Systems: David Lees, Leslie Keeley, Ted Shab, Kim Hubbard, Dennis Heher, Tom Dayton, Marleigh Norton, Jay Trimble, Paul Backes (JPL), Antonio Diaz-Calderon (JPL)

Mars Science Consultants: Nathalie Cabrol, Gloria Hovde

Mission Simulation Facility: Greg Pisanich, Lorenzo Flueckiger, Laura Plice, Michael Wagner, Chris Neukom, Eric Buchanan.