# Stereo Perception and Dead Reckoning for a Prototype Lunar Rover

Eric Krotkov, Martial Hebert, and Reid Simmons

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
epk@cs.cmu.edu

## Abstract

This paper describes practical, effective approaches to stereo perception and dead reckoning, and presents results from systems implemented for a prototype lunar rover operating in natural, outdoor environments.

The stereo perception hardware includes a binocular head mounted on a motion-averaging mast. This head provides images to a normalized correlation matcher, that intelligently selects what part of the image to process (saving time), and subsamples the images (again saving time) without subsampling disparities (which would reduce accuracy). The implementation has operated successfully during long-duration field exercises, processing streams of thousands of images.

The dead reckoning approach employs encoders, inclinometers, a compass, and a turn-rate sensor to maintain the position and orientation of the rover as it traverses. The approach integrates classical odometry with inertial guidance. The implementation succeeds in the face of significant sensor noise by virtue of sensor modelling, plus extensive filtering.

The stereo and dead reckoning components are used by an obstacle avoidance planner that projects a finite number of arcs through the terrain map, and evaluates the traversability of each arc to choose a travel direction that is safe and effective. With these components integrated into a complete navigation system, a prototype rover has traversed over 1 km in lunar-like environments.

# Table of Contents

# 1. Introduction

The lure of the Moon is strong—and humans are once again responding to the challenge. One promising, near-term scenario is to land a pair of rovers on the Moon, and to engage in a two-year, 1,000 kilometer traverse of historic sights, including Apollo 11, Surveyor 5, Ranger 8, Apollo 17 and Lunokhod II [15]. In this scenario, one of the rovers will be driven, returning continuous live video, while the other rover remains stationary, to provide high-resolution imagery of the surroundings and to serve as a communications link with Earth for both rovers. The rovers would periodically switch roles, continuing this way to explore the Moon's surface. The resulting experience is intended to attract mass participation and evoke strong public interest in lunar exploration [18].

While the hardware aspects of such a mission are daunting—including challenges in communications, power generation, thermal control, and rover reliability—the software control aspects are equally challenging. In particular, capabilities are needed for driving the rover over varied terrain and safeguarding its operation. Previous experience with planetary robots (in particular, Lunokhod II and the Viking soil scoop [9]) illustrated how laborious and unpredictable time-delayed teleoperation is for remote operators. Potentially superior modes of operation include supervised teleoperation and autonomous control, because in these modes the rover itself is responsible for making many of the decisions necessary to maintain progress and safety.

We have begun a program to develop and demonstrate technologies to enable remote, safeguarded teleoperation and autonomous driving in lunar-like environments. The aim is to provide both the techniques and evaluations of their effectiveness and reliability, in order to enable mission planners to make informed cost/benefit tradeoffs in deciding how to control the lunar rovers. To date, we have concentrated on local obstacle avoidance for autonomous operation, and have demonstrated a navigation system that uses stereo vision to drive a prototype lunar rover at an average speed of 10 cm/s over a kilometer of outdoor, natural terrain on an undulating plateau featuring sheer cliffs, mounds, and ridges. To date, our longest contiguous run has been 1,078 m, where 94% of the distance was traversed in autonomous mode and the rest in direct teleoperation. To our knowledge, this is a record distance for autonomous cross-country driving of a vehicle using stereo and only general-purpose processors (in our case, SPARC 10).

Our continuing research strives to increase the distance travelled and to decrease reliance on human operators. In 1995, we plan to demonstrate safeguarded teleoperation of up to 10 km, while increasing the complexity of the terrain traversed and the amount of time delay introduced in operating the rover. In 1996, we plan to achieve a 100 km traverse over a course representing the full variety of terrain to be encountered on the lunar mission.

This paper concentrates on two of the critical components of a multi-kilometer navigation system: stereo perception, and dead reckoning. To provide the context for these two components, we also describe the overall navigation system and some of its other key components. However, the purpose of this paper is to present a thorough treatment of the stereo and positioning components, not to provide a detailed account of an integrated system.

The principal contributions of the work reported here are in (1) the advanced development of a number of existing but disparate approaches into practical algorithms that are effective in natural terrain; and (2) the real-world demonstration and evaluation of these algorithms in field trials processing thousands of image pairs and millions of state sensor readings.

The next section reviews related work in stereo perception, dead reckoning, and navigation systems. Section 3 describes the rover currently used for our experiments. Section 4 provides an overview of the integrated navigation system, which in addition to stereo and dead reckoning includes an obstacle avoidance planner and command arbiter. Sections 5 and 6 describe the stereo perception and dead reckoning approaches that have been implemented, and present quantitative experimental results. Finally, we summarize the results, and address work required to return to the Moon in this millennium.

## 2. Related Work

### 2.1 Stereo Perception

A majority of the efforts to navigate mobile robots in natural terrain have employed laser rangefinders [4][12][13][20] or proximity sensors [1] rather than stereo. There have been notable exceptions. Matthies [23] developed a near-real-time system using Datacube hardware, and demonstrated 100 m traverses with a planetary rover prototype. Faugeras et al. [3][27] developed a real-time system using Digital Signal Processors (DSPs). Ross [30] developed a trinocular stereo system for the Dante I walking robot. Kanade [14] developed a video-rate stereo machine that is capable of generating a dense range map at 30 frames per second.

Each of these stereo systems has its virtues, and each exhibits great promise. Our contribution, which has not been reported in the literature, is an approach that achieves good performance without either special-purpose hardware or stringent requirements for alignment.

### 2.2 Dead Reckoning

Dead reckoning includes odometry, inertial guidance, and other “self-contained” sensing. The fundamentals of odometry (relating wheel revolutions to vehicle pose) and inertial guidance (relating angular accelerations to vehicle pose) are well known [7], so we will not discuss these.

Our contribution is to integrate odometry and inexpensive inertial guidance. We use a suite of simple, inexpensive sensors (wheel encoders, compass, turn-rate sensor, and inclinometers) which are not standard equipment on most mobile robots. As a consequence of the choice of inexpensive sensors, our approach requires aggressive filtering of the sensor readings.

### 2.3 Navigation Systems

A significant number of systems have been fielded that are capable of self-reliant operation in outdoor, natural terrain. These include land vehicles such as the Navlab [34] and air vehicles such as the cruise missile. However, these vehicles operate under constraints (on mass, power, etc.) far different than those faced by planetary rovers.

Restricting attention to planetary rovers, we know of only six systems other than our own that have achieved rough terrain traverses approaching or equalling 100 meters under control modes other than teleoperation.

- Robby is a six-wheeled rover developed at the Jet Propulsion Laboratory (JPL). Robby used stereo and a semi-autonomous control mode to traverse 100 m in a desert arroyo [23]. For this traverse, operators specified navigational waypoints, between which the vehicle traveled autonomously.

- Rocky is a family of six-wheeled rovers developed at JPL. The Rocky rovers use a control architecture that creates task-oriented modules defined as “behaviors” and provides for programmable inter-module connections [8]. Rocky rovers have performed a number of traverses in outdoor terrain using short-range sensors such as laser light stripers. The Rocky concept is currently the baseline design for the Mars Pathfinder mission, calling for tens of meters of travel, and scheduled for launch in 1996 [33].
- Adam is a six-wheeled rover developed as part of the Iares project in Europe, whose goal is to build a ground demonstrator with capabilities to perform an ambitious (1,000 km traverse over 13 months) scientific mission to Mars by the end of the millennium [10]. The approach has been tested in an experimental testbed called Eden [4]. In Eden, Adam used vision for landmark recognition, a laser rangefinder for obstacle detection, a coarse-scale planner for sub-goal selection, and 2D and 3D motion planners for obstacle avoidance. Adam executed autonomously a “Go To(Landmark)” task, where the landmark is a known artificial object, in an unknown environment that is gradually discovered by the robot. Although the degree of autonomy is high, the system has yet to be tested in extreme terrain for long durations.
- Ambler is a six-legged walker developed at Carnegie Mellon University [2]. Under autonomous control, Ambler traversed over 100 m in rugged terrain including meter-scale obstacles, and over 500 m in benign terrain including 15 degree slopes and cross-slopes [21]. The navigation system relied on accurate terrain maps built from laser rangefinder data.
- Dante II is an eight-legged walking and rappelling robot developed at Carnegie Mellon University [18]. Dante II descended 400 m over 35 degree slopes into the crater floor of Mt. Spurr, an active volcano in Alaska, where it successfully collected gas samples. The navigation system employed a number of control modes, ranging from direct teleoperation to full autonomy, using cameras for visual feedback and a laser rangefinder for terrain mapping.
- The “SSV-B” vehicle is a modified HMMWV developed at Martin Marietta under the Unmanned Ground Vehicle program. This vehicle traversed on the order of 100 meters of cross-country terrain using stereo vision for obstacle avoidance at a speed of 2 meters/sec ADD CITATION. In a previous effort, the CARD-II system demonstrated a version of safeguarded teleoperation [22].

The accomplishments of these navigation systems are remarkable and historic. One distinguishing feature of the present work is greater travel distance (1 km, to date). Another distinguishing feature is that the other systems, except Robby, rely on laser rangefinders or proximity sensors rather than on stereo vision. The main difference with the Robby system is our utilization of general-purpose computing hardware.

## 3. Mobile Platform

Until a new lunar rover [15] enters service, we are using a vehicle designed and built by Sandia National Laboratories [28] as a testbed to develop the remote driving techniques needed for a lunar mission. Ratler (Robotic All-Terrain Lunar Exploration Rover) is a battery-powered, four-

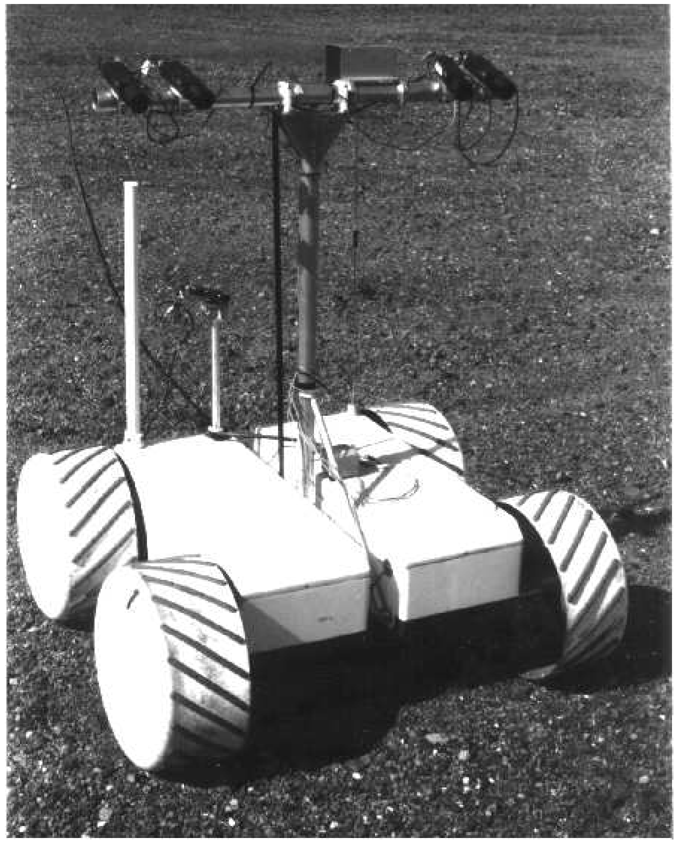

**Figure 1** Ratler

wheeled, skid-steered vehicle, about 1.2 meters long and wide, with 50 cm diameter wheels (Figure 1).

Ratler is articulated, with a passive axle between the left and right body segments. This articulation enables all four wheels to maintain ground contact, even when crossing uneven terrain, which increases Ratler's ability to surmount terrain obstacles. The body and wheels are made of a composite material that provides a favorable strength-to-weight ratio.

Sensors on Ratler include wheel encoders, a turn-rate gyro, a compass, a roll inclinometer, and two pitch inclinometers (one for each body segment). There is one color teleoperation camera, and we have added a camera mast and four black and white cameras for stereo vision (only two of which are used for the results reported here).

On-board computation is provided by 286 and 486 CPU boards, connected by an STD bus. The bus also houses a digital i/o board, an analog to digital conversion board, a 4-channel encoder board, and a pair of digitizer boards for the stereo cameras. Currently, the on-board computers handle servo control of the motors, sensor data acquisition, and communication with the off-board computers. Communications take place over a 4800 baud data link and a 2.3 GHz microwave video link. Off-board computers, which run the stereo, planning, and user interface modules, consist of several Sun workstations.

## 4. Navigation System

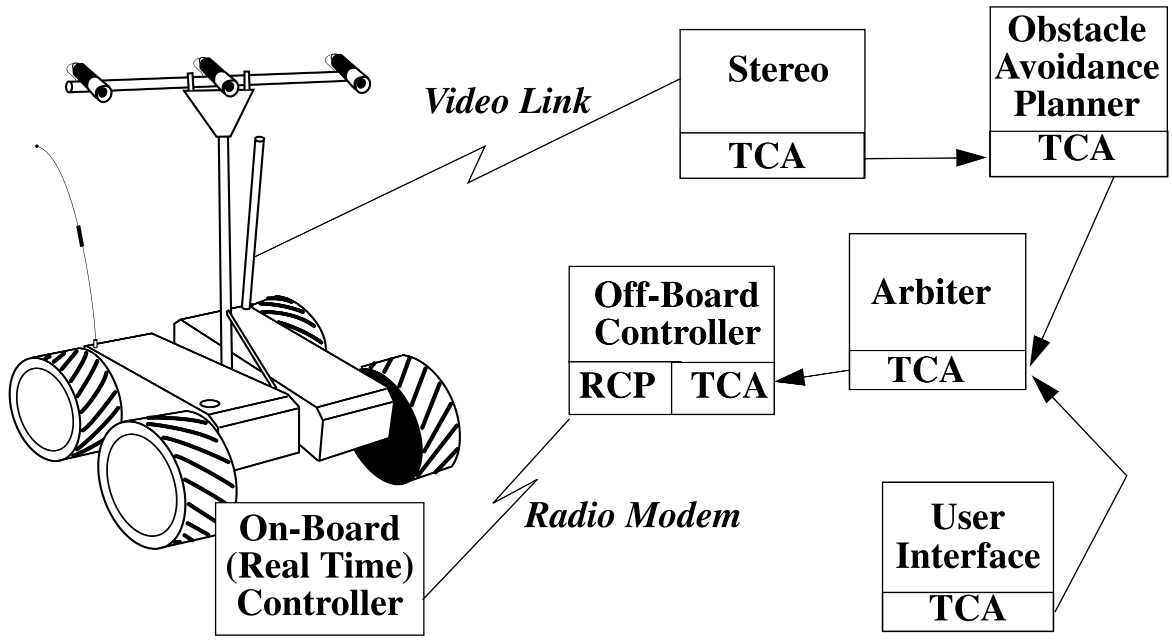


**Figure 2** Navigation system

Figure 2 presents a block diagram of the overall navigation software architecture. As described in Section 3, it is divided into on-board and off-board components, communicating via two radio links for video and data. Eventually, many of the off-board processes will be moved on-board the new lunar rover. The basic data flow is that the stereo process produces terrain elevation maps which are passed to the obstacle avoidance planner, which uses them to evaluate the efficacy of traveling along different paths. In safeguarded teleoperation mode, these recommendations are combined with the desires of a human operator to choose the best path to traverse (in autonomous mode, there is typically no operator input). The command arbiter then forwards steering and velocity commands to the off-board controller, which packages and ships them, using the RCP protocol developed at Sandia, to the on-board controller, which then executes the commands and returns status and sensor information. The off-board controller also performs much of the position estimation task and provides some safety mechanisms, such as stopping the rover if roll or pitch inclinometers exceed safe values. All components operate concurrently and receive their inputs asynchronously.

We believe that this arbitration architecture facilitates combining the best features of human and machine control: The human operator provides high-level guidance as to where the rover should travel (minimizing the need for global path planning by the rover), and the rover provides safeguarding by detecting and avoiding imminent hazards (which increases overall safety, and can reduce the level of fatigue and frustration typically experienced by operators in time-delayed, remote driving tasks).

In the four following subsections, we describe briefly the components of the system that are not the principal subject of this paper: the command arbiter, the obstacle avoidance planner, the user interface, and the approach to system integration.

## 4.1 Arbiter

The command arbiter asynchronously accepts path evaluations from various sources and chooses the best steering angle based on those evaluations. Each path evaluation consists of a steering angle, value, and speed. If the value is “veto” then that steering angle is eliminated from consideration. Otherwise, recommendations for that steering angle from all sources are combined using a weighted sum. This produces a new set of evaluations that combines all the sources of input.

The arbiter then finds, within this combined set of evaluations, the largest contiguous set of steering angles whose values are all within 90% of the maximum value, and chooses the midpoint of that set as the commanded steering angle. The speed chosen is the minimum speed recommended by all the sources. The idea is to prefer wide, easily traversable areas over directions that might be a bit more traversable, but have less leeway for error if the rover fails to track the path precisely.

The path evaluations are also tagged with a robot pose. If the tagged pose differs significantly from the rover’s current dead-reckoned pose, then those path evaluations are invalidated. If the evaluations from all the sources are invalidated, then the arbiter issues a command to halt the rover. In this way, the arbiter safeguards against other modules crashing, or otherwise failing to provide timely inputs. Similarly, we guard against failure of the arbiter itself by having it send commands of the form “steer in this direction for X meters.” The controller stops the robot if it does not receive a new command before that distance has been traversed.

## 4.2 Obstacle Avoidance Planner

To decide where it is safe to drive, we have adapted techniques developed in ARPA’s Unmanned Ground Vehicle program for cross-country navigation [16]. The basic idea is to evaluate the hazards along a discrete number of paths (corresponding to a set of steering commands) that the rover could possibly follow in the next few seconds of travel. The evaluation produces a set of “votes” for each path/steering angle, including “vetoes” for paths that are deemed too hazardous to traverse. In this way, the rover steers itself away from obstacles, such as craters or mounds, that it cannot cross or surmount.

The obstacle avoidance planner first merges individual elevation maps produced by the stereo system to produce a more complete terrain map of the area up to seven meters in front of the rover. Map merging is necessary because the limited fields of view of the cameras do not allow a single image to view sufficient terrain. Map merging is done by using the dead-reckoned pose information (Section 6) to transform stereo range maps that are in camera coordinates into global coordinates. The range points are then discretized into 25 cm square grid cells, and the average deviation between the existing terrain map and the new map is subtracted out, to compensate for systematic differences between range maps.

To save computation time, only a small segment of the stereo image is requested, at reduced resolution (typically only about 2% of the total available image). The planner dynamically chooses which portion of the image the stereo system should process, based on the current vehicle speed and expected cycle time of the perception/planning/control loop. The idea is to look far enough ahead to see three vehicle widths in the cameras’ fields of view, and to process enough of the image so that there is some overlap for doing map merging, but also to limit the overlap so that the stereo and planning cycles are approximately the same (since the processes are concurrent, this implies that the planner will not have to wait for the stereo component, but can operate continuously).

To evaluate the potential steering commands, the planner uses a detailed model of the vehicle's kinematics and dynamics to project the expected path of the rover forward in time over the terrain. This produces a set of paths, one for each potential steering direction. By examining the terrain map elevations beneath the wheels at each point along a path, the planner can estimate the expected rolls and pitches along the path. If the maximum (absolute) roll or pitch of a body segment exceeds an allowable threshold, or if the path crosses a significant area that has not yet been imaged by the cameras, then the path is vetoed. Otherwise, the "value" of that path is a weighted sum of the roll angle, the pitch angles, and the percentage of underlying terrain that is known. The path evaluations are sent to the arbiter module, along with the desired travel speed and current dead-reckoned pose of the vehicle. Currently, on a SPARC 10, the obstacle avoidance planner takes about 500 msec per cycle (map merging and path evaluation).

### 4.3 User Interface

The other source of path evaluations is human input via a graphical user interface. Note that the human operator does not control the rover directly: all commands are routed through the command arbiter. This facilitates switching between the different modes of operation: direct teleoperation, safeguarded teleoperation, and autonomous navigation.

The current user interface consists of an "electronic joystick," which utilizes the computer mouse to command the robot's direction and speed, and a number of indicators, both textual and graphical, that indicate pertinent information such as commanded and instantaneous robot speeds, roll and pitches, position, and status. Visualization of the terrain is provided by a color camera mounted toward the rear of the rover, which is transmitted to a monitor over the microwave radio link.

The operator indicates a desired steering angle and speed via the electronic joystick. The recommendations sent to the arbiter are determined by a Gaussian distribution centered at the chosen steering angle. The variance of the Gaussian dictates how much leeway the system has to deviate from the operator's intentions: If the variance is zero, then only one steering angle can be chosen (the rest are vetoed by the user interface); as the variance grows, more and more steering angles are given non-zero values, which means they could be chosen if the obstacle avoidance planner values them highly enough.

In direct teleoperation control mode, the operator's commands are passed through the arbiter unaltered. Direct teleoperation is necessary when the rover gets into situations where the safeguarding software would otherwise prevent motion. This mode is reserved for experienced drivers in exceptional situations.

In autonomous operation, operator input is ignored. In autonomous mode, the operator can specify a goal point to be achieved, but the local behavior of the robot is determined solely by the obstacle avoidance planner.

The third mode, safeguarded teleoperation, is seen as the standard way in which the lunar rover will be operated. In this mode, input from the human and the obstacle avoidance planner are combined by the arbiter: the operator presents a desired direction to travel, and the obstacle avoidance planner can veto it, causing the robot to refuse to travel in that direction. The idea is that the software safeguards should prevent the operator from damaging the rover, but not otherwise interfere with the control. In addition, if the human chooses not to provide input, the rover will navigate autonomously. In this way, operator fatigue can be reduced by letting the robot operate on its own when it is in benign terrain, while still enabling the human to take over control at any moment.

### 4.4 System Integration

The perception, planning, arbiter, user interface, and off-board controller processes all run on two Sun workstations, connected via Ethernet. Interprocess communication and task sequencing is performed by the Task Control Architecture (TCA). TCA is a general-purpose architecture for mobile robots that provides support for distributed communication via sockets, task decomposition and scheduling, resource management, execution monitoring, and error recovery [32]. TCA-based systems consist of a number of distributed, concurrent processes that communicate via coarse-grained message passing. TCA connects processes, routes messages, and coordinates overall control and data flow. TCA also provides capabilities to log and analyze message traffic, which has proven to be useful in resolving timing issues in the autonomous navigation system.

Concurrency is important in achieving the levels of performance needed (1-2 Hz cycle time is desirable). In particular, perception and planning functions run simultaneously: while the obstacle avoidance planner is using one stereo elevation map to evaluate paths, the stereo module is processing another image. When stereo completes, it asynchronously sends the image to the planner. Likewise, the arbiter receives asynchronous path evaluations from the user interface and obstacle avoidance planner, combining the most recent information to produce steering commands. Meanwhile, the controller module receives asynchronous commands and requests for dead-reckoned pose information from different modules. While it is admittedly more difficult to develop and debug distributed, concurrent systems, they have great advantages in achieving real-time performance.

## 5. Stereo Perception

The stereo perception system consists of a stereo module that derives terrain information from binocular images. The hardware consists of two CCD cameras, with auto-iris 8 mm lenses (42 deg vertical by 55 deg horizontal), mounted on a motion-averaging mast, a video link, and off-board frame grabbers and processors. The motion-averaging mechanism is a four-bar linkage; its kinematics are well-known, and are incorporated in a straight-forward fashion in the transformation from camera to vehicle coordinates. The stereo perception software takes as input a stereo pair and outputs arrays of the three coordinates $X$, $Y$, and $Z$ of the image pixels in the camera frame of reference.

The sets of points measured by stereo are accumulated over time into a terrain map centered at the vehicle which is used by the navigation system to drive the vehicle. Since all the data is accumulated in the terrain map, this system does not deal with dynamic scenes which would require explicit identification of moving objects.

### 5.1 Image Acquisition

We mounted the cameras on a mast to satisfy a number of imaging requirements. Here we describe three of the more important requirements.

1) *Lookahead distance*: The cameras must look far enough ahead to allow the robot enough time to stop or maneuver around an obstacle. The stopping distance is the sum of the distance traveled before braking and the distance traveled while braking:

$$d_{stop} = t_{stop}v + v^2 (2\mu g)^{-1}$$

where $t_{stop}$ is the braking time, $v$ is the velocity of travel, $\mu$ is the coefficient of sliding friction, and $g$ is gravitational acceleration. We estimate the braking time to be about 2 sec by summing the times required to acquire stereo imagery, compute stereo disparities from the images, and detect an obstacle from the stereo disparities. Assuming that the vehicle travels at the maximum velocity of 0.7 m/s, and a coefficient of sliding friction of 0.25, the stopping distance is 1.5 m from the front wheels. This stopping distance of 1.5 m constrains the camera height to be at least 66 cm, the height at which the line of sight just grazes the top of the wheels.

2) *Width of field of view*: The cameras must see at least 3 vehicle widths (about 400 cm), at all distances beyond the lookahead distance, so that the rover can maneuver around obstacles one vehicle width in size.

3) *Resolution*: An obstacle 20 cm tall must subtend at least 6 pixels in order to be reliably detected.

Given these requirements, we identified the key variables to be camera height, camera baseline, width of field of view, and tilt angle. We performed trade-offs on these variables. The analysis is complicated due to conflicting requirements. For example, raising the cameras increases the width of field of view (good), but decreases the resolution (bad). After extensive simulation and experimentation, we converged on a camera height of 1.5 m, a baseline of 0.9 m, and a tilt angle of 25 deg down from horizontal.

To maximize image stability as the rover traverses surface irregularities, we designed and built a motion-smoothing four-bar linkage that averages the pitch of the two Ratler bodies. This linkage has proven to be extremely valuable in providing reasonably overlapped images while traversing extreme terrain.

## 5.2 Stereo Matching

We denote by $x$ and $y$ the axis of coordinates of the image plane, $y$ being vertical. We assume in the stereo matching that the epipolar lines are the scanlines of the images so that, given a pixel $(x_r,y_r)$ in the right image, we need to search for the best matching pixel $(x_l,y_l)$ in the left image such that $x_l = x_r + d(x_r,y_r)$ and $y_l = y_r$, where $d(x_r,y_r)$ is the disparity at $(x_r,y_r)$. In order to ensure that the epipolar lines are correctly aligned with the scanlines, we use a rectification procedure developed by Robert [29]. The rectification is applied to the input images and all the algorithms described below are applied to the rectified images.

The computation time for rectifying a full image is 100 ms. As we will see below, we use typically only one third of the image. Since only the selected part of the image is rectified, the total time for rectifying both images of a stereo pair is on the order of 70 ms, a negligible fraction of the total computation time compared to the fraction taken by correlation.

The best disparity $d(x,y)$ is computed by finding the maximum over $d$ of the normalized correlation $C(x,y,d)$ [6]:

$$\frac{\dfrac{\sum_{(x,y)\in W} I_l((x+d),y)I_r(x,y)}{n} - \dfrac{\left(\sum_W I_l(x+d,y)\right)\left(\sum_W I_r(x,y)\right)}{n^2}}{\sigma_l \sigma_r}$$

In this expression, $W$ is the window $[x-w_x,x+w_x][y-w_y,y+w_y]$, $\sigma(x,y)$ is the standard deviation of the intensity values in $W$ and $n$ is the number of pixels in the window, $n = (2w_x+1)(2w_y+1)$. Since all the computations are referenced to the right image, we will simplify the notations by dropping the index $r$ and by denoting the pixel position simply by $(x,y)$. We will denote by $C(x,y)$ the correlation value for the best disparity $d(x,y)$.

In order to achieve disparity resolution better than the image resolution we use a parabolic interpolation that uses the correlation values of the two closest disparities.

We chose the normalized correlation criterion over the sum of squared differences (SSD) for two reasons. First, the normalized correlation does not need to use a LOG filter to remove photometric differences between the images. Second, the normalized correlation $C(x,y)$ provides a natural measure of confidence of the disparity value at pixel $(x,y)$. Moreover, the increase in computation time compared to the SSD criterion is minimal.

Area-based stereo is a well-established technique but it is known to produce a potentially large number of false matches due to lack of texture, occlusions, and repetitive patterns. It is especially important to be able to filter out those false matches in the context of navigation applications because even a few erroneous points out of hundreds of images may have negative or even catastrophic consequences.

In order to achieve the level of reliability required for navigation, we use four types of filtering. The first two use thresholds $\sigma_{min}(x,y)$ and $C_{min}(x,y)$ on the standard deviation of the distribution of intensity in the neighborhood of $(x,y)$ and on the best correlation at $(x,y)$, respectively. These classical filters eliminate the low-textured areas and part of the occluded areas.

The third filter is designed to eliminate ambiguous matches. It uses a threshold $M$ on the relative difference between the global maximum of correlation and the second best peak in the correlation peak, $C'(x,y)$. Specifically, a point is rejected if $(C(x,y) - C'(x,y))/C(x,y)$ is lower than $M$. This test is effective in discarding pixels at occlusion boundaries and ambiguous matches due to repetitive patterns. The last filter is a median filter on the disparity map.

Figure 3 shows a typical stereo pair from our test site, the corresponding rectified pair, and the disparity map. The disparity map is computed from 2 m to 14 m at full resolution although we will describe a more efficient use of the stereo matcher below. Except for the resolution, the parameters used for this result and for all the navigation experiments are as follows: 640 columns, 480 rows, $w_x = 25$, $w_y = 17$, $\sigma_{min} = 2$, $C_{min} = 0.5$, and $M = 10$ percent.

After the stereo matching, $d(x,y)$ is converted to a 3-D point $P = (X(x,y),Y(x,y),Z(x,y))$ by using the projective transformations $P_l$ and $P_r$ between the two images and a coordinate system referenced to the vehicle. $P_l$ and $P_r$ are computed by combining the rectification matrices with the calibration matrices computed using a standard calibration procedure [29]. This coordinate system in which the points are expressed is set up so that the $Z$ axis is up, the $Y$ axis is the direction of travel of the

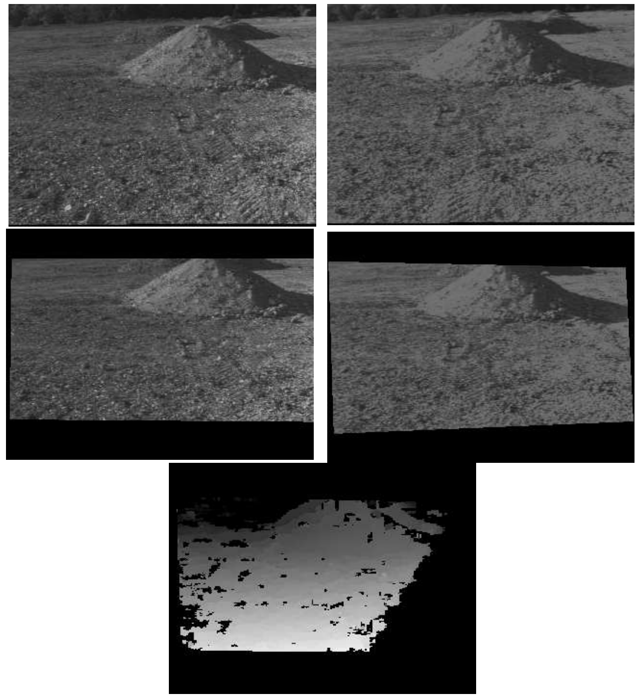

**Figure 3** Typical input stereo pair (top); Rectified images (middle); Disparity map (bottom)

vehicle, and the origin is at the base of the mast supporting the cameras. In the remainder of the paper, Cartesian coordinates are expressed with respect to this vehicle-based coordinate system.

We predicted the Cartesian coordinate errors based on a one pixel error in disparity using the current camera configuration. For targets from 0 to 20 meters, the $Y$ error increases quadratically, that is $\sigma_Y = KY^2$, where $\sigma_Y$ is the standard deviation of the error in $Y$. In our current setup, the error in $Y$ increases to almost 60 cm at 20 m, while the errors in $X$ and $Z$ do not exceed 6 cm. This suggests that our stereo gives a precision level that is comparable with the precision of laser range finders used in other navigation systems [12].

We employ this simple error model, based on a one pixel disparity error, for the purpose of verifying that the stereo setup generates “reasonable” errors (cf. Section 5.4). For the purpose of achieving smaller errors, sub-pixel interpolation techniques and more sophisticated uncertainty models have been shown to lead to much better than one pixel disparity error [24][25].

### 5.3 Window Selection

The previous section briefly described the stereo matching and filtering techniques that we use. In this and the following section, we describe improvements to the basic stereo algorithms needed to make them usable in a practical navigation application requiring computational efficiency, robustness, and precision.

In order to apply correlation-based stereo to a practical navigation system, we are faced with a difficult challenge. We need to process images at a speed high enough to sustain continuous motion of the vehicle while retaining maximum precision on the disparity estimates. Two ways of addressing this challenge have been proposed in the past. First, the image can be processed at a coarser resolution [23][30]. In this case, the processing time can be reduced arbitrarily by decreasing the resolution of the images until it matches the needs of the application. However, the quality of the resulting maps is degraded because the resolution of the disparity values decreases with the resolution of the image.

A second approach is to use special-purpose hardware in order to perform the correlations. This solution has led to several real-time and near real-time stereo systems using DSP [3] or Datacube systems [23]. However, for our application, considerations of cost, power, and availability limit us to conventional computing. Although some of the shortcomings of those solutions are specific to our applications, we believe that they apply to some extent to any practical vision system for which cost, speed, and precision are all issues of equal importance.

Our approach to this problem is based on the observation that an autonomous driving system needs to process only a small subset of the image, as long as that subset is selected judiciously as a function of the speed of the vehicle and of the geometry of the camera system. Intuitively, the vision system needs to process only the sub-image that adds information to the map built from all the previous images. As we will see, the size of the sub-image is quite small, assuming that the system is in steady state and that the speed is approximately constant. A complete theory supporting this observation was developed by Kelly [17]. A similar window selection algorithm is described in [11].

We exploit this property in the following way: the planner computes the interval $I_Y = [Y_{min}, Y_{max}]$ of distance from the vehicle within which it needs data in order to expand its map. $I_Y$ is computed from the speed of the vehicle and from the anticipated delay in getting the result from stereo [16].

The stereo module uses $I_Y$ to compute the bounds of a sub-image, $(y_{min}, y_{max})$, and the corresponding bounds in disparity $(d_{min}, d_{max})$. This computation requires some assumptions about the geometry of the environment because $I_Y$ constrains only the region of interest in the $(X,Y)$ plane but not along the $Z$ axis. Specifically, we assume maximum and minimum heights of the objects in the scene, $Z_{min}$ and $Z_{max}$. That is not to say that objects of larger size cannot be detected but that only the parts of the objects between $Z_{min}$ and $Z_{max}$ are computed, which is sufficient for navigation purposes.

The bounds are computed by first finding the $y$ coordinates of the pixels in the right image such that $x$=0 or $x = dim_x$ and $Y = Y_{\{min,max\}}$ and $Z = Z_{\{min,max\}}$. These points are at the boundary of the region of interest. The $y$ coordinates can clearly be computed by solving for $y$ and $X$ at each of the boundary points. The minimum and maximum values of the $y$ coordinates are the vertical bounds of the sub-image. Once their $y$ and $X$ coordinates are computed, the boundary points are also projected in the left image and the minimum and maximum differences between the $x$ coordinates of their projections in the left and right images are $d_{min}$ and $d_{max}$.

We have empirically measured the stereo computation time as a function of $\Delta Y = Y_{max} - Y_{min}$ for $Y_{min}$ = 5 m (Figure 4). The computation times indicated on the ordinate axis are normalized with respect to the largest value. The reduction in computation time compounds two effects: first, the size of the sub-image is reduced, thus decreasing the number of pixels processed; second, the disparity interval is reduced and, correspondingly, the number of steps in the correlation search. The computation time is not quite linear because of constant time initializations that must be performed before the actual stereo matching.

Figure 5 shows the average of the minimum and maximum value of the requested image rows, the disparity, and the corresponding $Y$ coordinates. These values were computed by averaging the

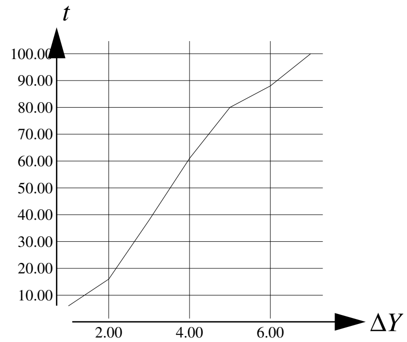


**Figure 4** Stereo computation time as function of the width in Y of the desired interval for Ymin = 5m

| | min | max | Δ |
|---|---|---|---|
| *y* (pixels) | 130 | 233 | 103 |
| *d* (pixels) | 44 | 118 | 74 |
| *Y* (meters) | 4.2 | 7.5 | 3.3 |

**Figure 5** Average values of disparity range, sub-image size, and distance interval recorded during typical run

values in the planning requests recorded during a 150 m (310 images) run of the system with the vehicle moving at 0.15m/s.

After an initial adjustment phase, the values in the planning requests remain within 10% of the average values shown in Figure 5. These values confirm that only a small fraction of the image, corresponding to a 3 m swath of terrain, needs to be processed once the navigation system is in a steady state. Thus, stereo is a viable solution provided that the correct sub-image selection is used.

### 5.4 Partial Subsampling

Further reduction of the computation time may be achieved by observing that it is not necessary to process the data at full resolution. More precisely, the resolution on the ground of the data points obtained after transformation is too high compared to what is actually needed for evaluating terrain traversability. For example, the distance in *X* between consecutive points on a scanline at full resolution at 10 m is on the order of 5 cm, whereas a 25 cm grid is typically sufficient for evaluating navigability. This suggests that it is advantageous to subsample the image in order to process fewer pixels while retaining enough data density for navigability evaluation.

This idea has been used successfully in navigation systems using laser range finders [16]. In the case of stereo, we have to be more cautious because simple subsampling will automatically degrade the resolution of the disparity. In other words, we want less data but not at the cost of less accurate data.

We modified the correlation algorithm to get the best compromise between speed and precision by using a partial subsampling in which the disparity is evaluated on a subsampled set of pixels but in which the disparity at a given pixel is searched by using the maximum resolution. We call this approach partial subsampling.

Let $\delta x$ and $\delta y$ be the sampling steps in columns and rows respectively, i.e., the disparity is computed only at the pixels $(x_r, y_r)$ such that $x_r = k\delta x$ and $y_r = k\delta y$. We assume that the window sizes $w_x$ and $w_y$ are multiples of $\delta x$ and $\delta y$, respectively. The window $W$ over which the sums are taken in $C(x_r,y_r,d)$ is now defined as:

$$W = \{ (x, y) \in W_f | \left( x = x_r + k\delta x \right), \left( y = y_r + k\delta y \right) \}$$

where $W_f$ is the full resolution window:

$$W_f = [x_r - w_x, x_r + w_x] \times [y_r - w_y, y_r + w_y]$$

In the expression of $C(x_r,y_r,d)$, only the sum of products has to be recomputed for each disparity since the mean and standard deviation at each pixel do not depend on $d$ and therefore can be computed only once. For each disparity $d$ in $[d_{min},d_{max}]$, the products $I_l(x_r+d,y_r)I_r(x_r,y_r)$ are computed for all the subsampled values of $(x_r,y_r)$. Let us denote the sum of products at $(x_r,y_r)$ over $W_f$ by $S(x_r,y_r)$, and the sum of products over column $x_r$ only by $S_y(x_r,y_r)$:

$$S\left( x_r, y_r \right) = \sum_{(x, y) \in W} I_r (x, y) I_l (x + d, y)$$

$$S_y\left( x_r, y_r \right) = \sum_{y \in \left[ y_r - w_y, y_r + w_y \right]} I_r\left( x_r, y \right) I_l\left( x_r + d, y \right)$$

Then $S(x_r+\delta x,y_r)$ is computed recursively from $S(x_r,y_r)$, $S_y(x_r-w_x,y_r)$, and $S_y(x_r+w_x+\delta x,y_r)$ as

$$S\left( x_r + \delta x, y_r \right) = S\left( x_r, y_r \right) + S_y\left( x_r + w_x + \delta x, y_r \right) - S_y\left( x_r - w_x, y_r \right)$$

This shows that, for a given disparity $d$, the map $C(x_r,y_r,d)$ can be computed recursively at the reduced resolutions $(\delta x,\delta y)$, thus reducing the computation required by $\delta x.\delta y$. At the same time, the resolution in disparity is maintained by computing the $C(x_r,y_r,d)$ for all the values of $d$, without subsampling it. Therefore, the partial subsampling, that is, subsampling the image without subsampling the disparity, does achieve our goal of more efficient stereo matching without loss of precision.

We conducted a series of experiments in order to verify that the precision is not affected by the subsampling. In those experiments, we placed boxes at distances ranging from 2 m to 8 m from the center of the vehicle by increments of 1m. The front faces of the boxes are parallel to the $X$ axis. We then ran the stereo program for the nine possible combinations of subsampling given by $\delta x = \{1,3,5\}$ and $\delta y = \{1,2,4\}$. At a given pixel in the image, we can plot the RMS error, $E_Y$, between the measured $Y$ values on the boxes and the true value. The true value is known exactly since we chose the test points on the target boxes.

Figure 6 plots $E_Y$ as a function of $Y$ for all nine pairs $\delta x = \{1,3,5\}$ and $\delta y = \{1,2,4\}$ and for $Y$ ranging from 1 m to 8 m by increments of 1 m. (We feel that it is important to show all nine plots in order to properly illustrate the result of the experiment. As a result, it is not possible to label individual graphs.) These plots show that the error in $Y$ is independent of $\delta x$ and $\delta y$. More precisely, except for random variations, there is no systematic difference between the error profiles for different levels of sampling. Those plots also show that the error in $Y$ computed from experimental data is

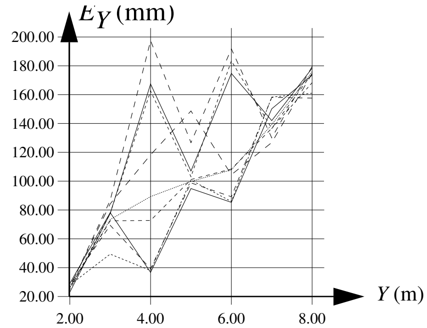


**Figure 6** Error in measured distance as a function of distance for different levels of subsampling

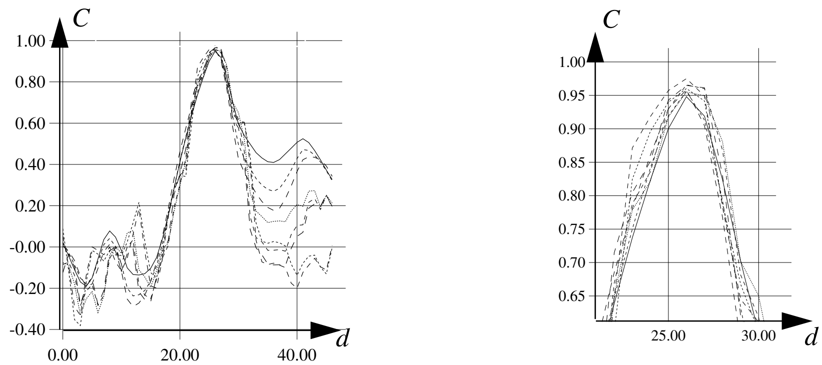


**Figure 7** Correlation curves at one pixel for the nine different cominations of levels of subsampling $\delta x = \{1,3,5\}$ and $\delta y = \{1,2,4\}$; the left plot is the complete correlation curve; the right plot is a magnified subset of the complete curve around the maximum

consistent with the theoretical error derived in Section 5.2. For example, the theoretical error at 8 m is 10 cm, while the actual error is 15 cm.

Another way to verify that the correct disparity is computed independently of the sampling values is to look at the shape of the correlation curve at one pixel for different values of sampling. Figure 7 plots the correlation values at one pixel as a function of disparity for the nine different combinations of $(\delta x, \delta y)$ for $\boldsymbol{\delta x} = \{1,3,5\}$ and $\boldsymbol{\delta y} = \{1,2,4\}$ at one pixel. The right part of Figure 7 shows the same curves in the neighborhood of the maximum correlation. This figure shows that the best disparity is the same at all resolutions and that there is little variation of the correlation values across resolution levels around the correlation peak, although as one would expect, there can be large variation far from the peak. In contrast, for $\delta x = 5$, the disparity computed using a conventional subsampling technique can be in error by as much as 5 pixels.

Although there is no loss in range resolution at a given pixel, using lower resolution does have an impact on the detectability of obstacles: Obstacles of width smaller than the sampling value may be missed because of the subsampling of the correlation window and of the range image. Figure 8 shows the width in millimeters as a function of the *Y* distance from the object to the camera for five different values of $\delta x$.

In practice we use fixed subsampling factors of $\delta x = 5$ and $\delta y = 4$. With these values, the stereo matching takes 0.7s on average on a Sparc10 workstation using the parameters of Figure 5. Those computation times are similar to those reported in [26].

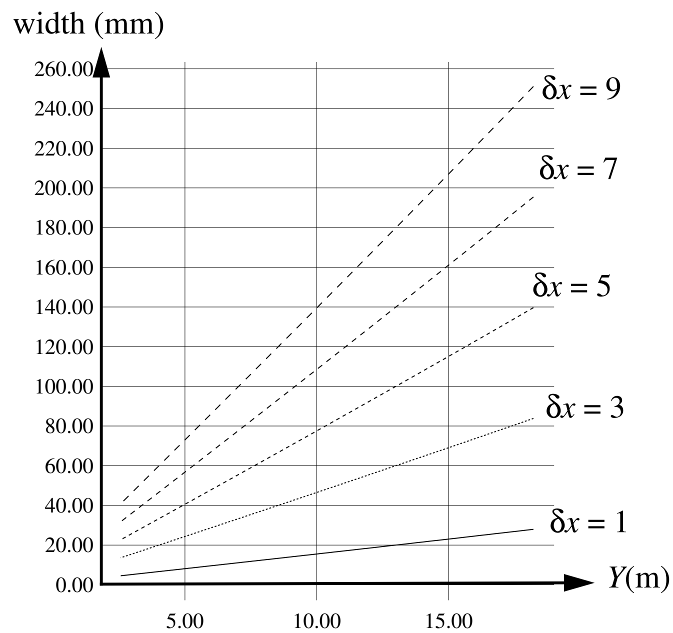


**Figure 8** Obstacle width in millimeters as function of the distance to the camera for different values of the horizontal sampling $\delta x$

### 5.5 Performance

The combination of selective windowing and partial subsampling allows us to achieve both the computational speed and the precision required for continuous motion at low speeds using a general-purpose workstation (with no special-purpose hardware).

The longest stereo run took place over 6 hours of intermittent operation (interrupted by rain and battery recharges) as the rover traversed 1,078 m over the rough terrain of a slag heap (Figure 9). During this trial, the stereo module processed over 3,000 image pairs, and computed at least $1.5\text{x}10^6$ three-dimensional points (based on the average values reported in Figure 5).

It is difficult to quantify the accuracy of the maps over the course of kilometer-long runs, because we do not know ground truth. Whenever we performed spot checks of the computed coordinates, they were correct within the precision of our evaluation of ground truth using a tape measure and a large-scale protractor. The only failures we observed were due to transient effects caused by disconnection of video cables, and by abrupt lighting changes that overwhelmed the auto-iris lenses.

## 6. Dead Reckoning

Knowledge of the position and orientation (pose) of the lunar rover is a fundamental requirement for a wide variety of entertainment and scientific missions. Position estimation on the Moon is a challenging problem because there are no artificial beacons, either active or passive, either on land or in orbit, to serve as navigational aids. (It is conceivable that a lunar rover could use signals from satellites in Earth orbit to fix its position. However, this scheme remains to be proven a viable alternative, and even it were, dead reckoning would still be valuable as a redundant and complementary source of information.)

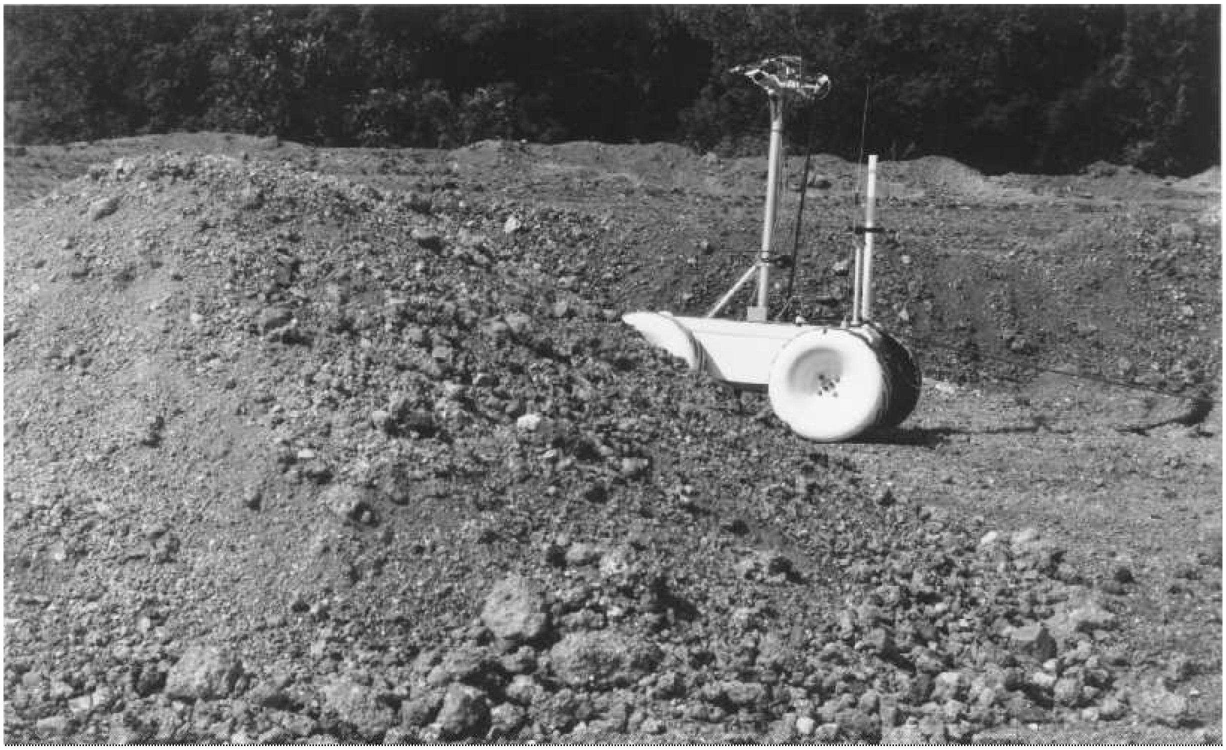

**Figure 9** Ratler at slag heap

For our purposes, it is important to determine quantitatively the rover location. This is important in many mission scenarios, but particularly for the one described in Section 1, which involves long-distance navigation to find specified sites. Thus, qualitative navigation techniques that permit travelling from landmark to landmark are not directly applicable.

The goal of the work reported here is to develop and demonstrate lunar-relevant position estimation capabilities with our mobile platform. The principal contribution is to develop and evaluate quantitatively a practical, multi-sensor dead reckoning system.

This system is implemented as a suite of algorithms for maintaining an estimate of the robot's position and orientation in a fixed, external reference frame. Currently, the bulk of the processing is performed by the off-board controller, where it is simpler to develop and debug. Once we have selected the algorithms and their parameters, we then port them to the on-board controller.

### 6.1 Sensors

Ratler includes four types of state sensors:

- Encoders on the 4 drive motors
- Inclinometers measuring the pitch of the left body, the pitch of the right body, and vehicle roll
- A turn-rate gyro
- A flux-gate compass. This device is not lunar-relevant since the magnetic field on the Moon is quite weak. However, devices such as north-seeking gyros can provide

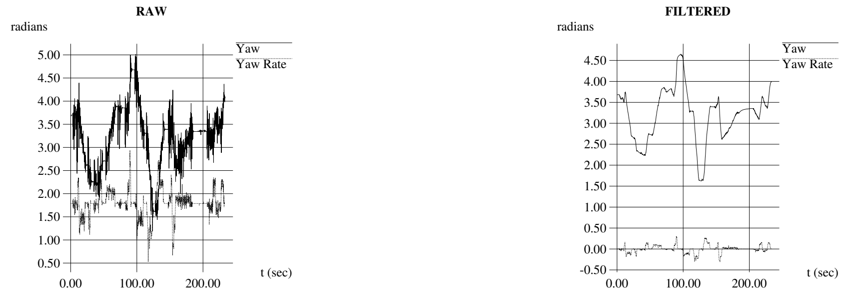


**Figure 10** Raw data (left) and filtered data (right)

comparable functions on the Moon, so it is not unreasonable to use a compass on Earth. The outputs of the sensors are digitized to 10 bits.

## 6.2 Filtering

The raw data from each of the four types of sensor is biased and noisy and sometimes corrupt. Thus, we have found it necessary to develop filters for preprocessing the sensor data. As an example, consider the yaw data provided by the compass. We omit detailed discussion of the filtering performed on the pitch, roll, and yaw-rate data, since we follow the same approach as for the yaw data.

In steady state, the compass signals are corrupted by random noise. When the rover accelerates, the compass signals are corrupted not only by random noise, but also by electrical noise and transients caused by dynamic effects.

We performed a spectral analysis of the data, and observed a cut-off frequency of 0.25 Hz. We implemented several low-pass filters, including Butterworth and Bessel filters. These were extremely effective in suppressing the noise, but also introduced a 2-3 cycle delay between the filtered value and the signal.

We also implemented a Kalman filter

$$\hat{x}_{n+1} = A\hat{x}_n + K\left(z - A\hat{x}_n\right)$$

with a sensor model in the matrix A that heavily weights the previous compass reading if the turn-rate sensor indicates that the robot is accelerating. In colloquial terms, this filter does not believe the compass when the rover turns. Figure 10 illustrates the results of applying this filter to the compass (yaw) and turn-rate (yaw-rate) sensors. The improvement in precision is evident and dramatic. We did not quantify the improvement in accuracy, because of the difficulty of acquiring ground truth measures of yaw and yaw-rate.

This type of adaptive filtering provides nearly optimal estimation with adaptive gains that depend on differences between expected and actual measurements, estimated errors in the previous navigation state and measurement uncertainties. The Kalman filter is well-suited for combining measurements, with the appropriate weightings, from sources that have different error statistics. Essentially, the filter we implemented estimates yaw integrating turn-rate sensor data and using compass data to limit yaw drift error.

Although the Kalman filter provides excellent results, it also introduces considerable computational overhead. The simpler low-pass filters gave comparable results except for the delay, which is not significant at typical rover speeds. Therefore we decided to use a second-order Butterworth filter for the compass data.

### 6.3 Position Estimation

In classical odometry, the new position and orientation of the robot is derived from the motions of the left and right wheels. Let the incremental displacement $\delta$ of the robot's center be the average of the distance travelled by the left and right wheels:

$$\delta = \frac{\delta L + \delta R}{2}$$

Let the incremental change in yaw be the ratio of the difference in wheel travel to the wheelbase:

$$\gamma = \frac{\delta R - \delta L}{B}$$

where $B$ is the distance between the wheels. The new position is given by

$$\begin{aligned} x_{n+1} &= x_n + \delta_{n+1} \cos\gamma_{n+1} \\ y_{n+1} &= y_n + \delta_{n+1} \sin\gamma_{n+1} \end{aligned}$$

Note that in this classical formulation it is not possible to compute the altitude. This is a serious limitation for traversal of terrain with relief. Also note that the new orientation is derived from the wheel travel distance, which is likely to be corrupted by slippage, skidding, and related effects.

To eliminate these two problems, our approach integrates classical odometry with inertial measures. Given the previous state and the new filtered state sensor readings, the new dead reckoning algorithm computes the current state of the robot. Let $\theta$ represent the filtered yaw, let $\phi$ represent the mean of the filtered readings from the left and right pitch inclinometers, and let $\Delta$ represent the mean of the differences between the current and previous encoder reading, expressed in metric units rather than counts. Then the new position is given by

$$\begin{aligned} x_{n+1} &= x_n + \Delta \cos\theta \cos\phi \\ y_{n+1} &= y_n + \Delta \sin\theta \cos\phi \qquad (1) \\ z_{n+1} &= z_n + \Delta \sin\phi \end{aligned}$$

This formulation incorporates implicitly the yaw-rate data through the filtering stage. Future work will extend (1) to include an integrated yaw-rate term. Future work will also explore separating rather than averaging the left and right pitches.

### 6.4 Results

A number of test runs were performed and analyzed in order to determine the accuracy of the dead reckoning system. These are summarized in Figure 11.

Trials A-D were performed on an outdoor slag heap, and for these we repeatedly commanded the robot to travel straight, both forward and backward, during which sensor readings were logged. Periodically we obtained a ground truth pose by stopping the vehicle and measuring its position and orientation using a tape measure and protractors. Trial A used the earliest, simplest generation

| Trial | Mean Distance | StDev Distance | Mean Orient. | StdDev Orient. |
|---|---|---|---|---|
| A | 17.0 | 2.0 | 0.0 | 3.7 |
| B | 3.1 | 1.3 | 0.5 | 11.0 |
| C | 0.7 | 1.9 | 1.7 | 22.0 |
| D | 0.1 | 1.2 | 0.5 | 11.0 |
| E | 2.7 | 2.1 | --- | --- |
| F | 5.3 | 6.6 | --- | --- |
| G | 2.5 | 2.3 | --- | --- |

**Figure 11** Statistics of position and orientation errors, in units of percent error

of the positioning system, in which the average of the two rear encoders (converted to linear feet) were used directly for $\Delta$, the immediate unfiltered compass reading was used for $\theta$, and $\phi = 0$ in (1). This trial yielded poor positional accuracy as indicated by the 17% error.

In response to, and guided by, these poor results, a number of constants in the low-level controller were changed, and the runs were repeated. The data collected during the second set of runs were used for the analyses shown in Trials B through D.

Trial B shows the results after the tuning of those controller constants, again using the simplest positioning system. The improvement in positional accuracy was dramatic. We observed a negligible bias in compass readings, but a larger standard deviation. Since none of our changes should have affected the compass readings, this increase is simply experimental "noise", probably attributable to the limitations in our ability to obtain the ground truth orientation using a protractor. However, these tests only reflect the compass error under "static" conditions, since the ground truth measurements were taken while the vehicle was not moving. (Armed with just protractors and tape measures, we did not have the resources necessary to obtain a continuous ground truth signal while the vehicle was in motion.) By plotting the compass readings during movement (see Figure 10), we found that the compass was quite noisy while the vehicle was in motion, but almost completely noiseless while the vehicle was stationary. Furthermore, we found the dynamic compass noise to be the most damaging error for the map merging step performed by the obstacle avoidance module. This prompted the development of two further techniques.

Trial C represents an attempt to eliminate the use of the compass altogether. In this trial, the encoder values of all wheels were used to detect turns and track both position and orientation. The large standard deviation of the orientation reveals that this provides the worst orientation estimates of all the trials.

For Trial D, the filtered compass data was used both to estimate the orientation and to update the x,y estimates using (1). The improvement in the estimate of orientation under dynamic conditions was dramatic (see Figure 10) and had a large positive influence on the map merging step of the obstacle avoidance module. Using the same data as with Trial B, we observed a notable improvement in the x,y position estimation as well. Again, the orientation error in Figure 11 reflects only a comparison of the estimate to ground truth under static conditions.

For Trials E, F and G, we repeated a series of tests, this time on a flat grassy field. This series of tests was undertaken to verify the previously observed results and to obtain data for trajectories other than forward and backward motions. In this case, the vehicle was manually steered through a closed-circuit course of distinguished locations whose positions had previously been measured. During these experiments, we did not collect a ground truth for vehicle orientation. Trial E shows the simplest positioning system (using encoders plus raw compass readings to update x,y). Trial F uses the same sensor data and the encoder-only positioning system. And Trial G uses the encoders and filtered compass data.

We draw the following conclusions from these trials and our experiences. First, the use of a compass helps considerably. While Trial C yielded favorable results, this was measured only on straight forward and backward motions. When the varied paths in Trial F were encountered, the use of encoders only were far less favorable. The compass's benefit is largely due to the fact that its readings are always derived with respect to an absolute orientation, so errors do not tend to accumulate when the vehicle turns. We also found the filtering of the compass data to be of great benefit for estimating the orientation while the vehicle was in motion and crucial for the map merging performed by the navigation system. The use of filtered compass data may have also improved the x,y position estimates somewhat, although this improvement was not reproduced in our final set of experiments. The pose estimates obtained using encoders and filtered compass data has been sufficient to demonstrate a successful 1,000 meter autonomous traverse in an outdoor slag heap.

However, better positional accuracies are needed, and the lack of improvement observed between Trials E and G suggests that further processing of the same sensor readings is unlikely to yield a significant improvement. We therefore also conclude from these experiments that further improvement requires alternative approaches, such as the utilization of visual landmarks.

## 7. Discussion

This paper has presented the navigation architecture for a prototype lunar rover, focusing on stereo perception and dead reckoning.

The navigation system uses a combination of on-board and off-board computation to control the vehicle, process stereo images, plan to avoid obstacles, and integrate machine and human recommendations regarding travel direction.

The stereo perception system consists of a camera head on a motion-averaging mast, which provides image pairs to a normalized correlation matcher. This matcher intelligently selects what part of the image to process (saving time), and subsamples the intensities (saving time) without subsampling disparities (which would reduce accuracy). In addition, the matcher uses a general rectification algorithm which permits the use of cameras in arbitrary configurations. The stereo system has operated successfully during long-duration field exercises, processing thousands of image pairs.

The dead reckoning system employs encoders, inclinometers, a compass, and a turn-rate sensor to maintain the position and orientation of the rover as it traverses. The system succeeds in the face of significant sensor noise by virtue of sensor modelling, plus extensive filtering.

Although both the stereo and positioning systems use classical sensors and previously developed algorithms, they have achieved unprecedented results, enabling long-duration (6 hours) and long-distance (1 km) outdoor traverses. The key contributions are in tailoring the general ideas to a specific robot performing a specific task, and in demonstrating practical and unprecedented performance.

Future work in stereo will continue to concentrate on robust, reliable operation in the face of occasionally abysmal sensor data. To enable better performance in avoiding obstacles, we will achieve a wider stereo field of view by replacing the binocular rig with a four-camera setup.

Future work in dead reckoning will improve performance by porting all of the preprocessing and processing onto the on-board system, thus reducing delays. We will then extend the positioning system to use visual landmarks; specifically, we plan to incorporate novel techniques for skyline navigation and celestial navigation (in the form of a digital sextant [5]).

Our experimental work will take two directions: we will continue to demonstrate and quantify autonomous navigation capabilities using stereo and local obstacle avoidance, and we will also investigate more carefully issues of mixed-mode and safeguarded teleoperation. This includes quantifying the performance improvements gained by adding various technologies, such as safeguarding and high-level commands.

To achieve the ambitious goals of the mission (1,000 km traverse over two years), we need more than these incremental improvements. We are currently investigating laser-based proximity sensors to provide coverage in front of the rover at ranges of one to two meters. Such sensors would be used to warn of impending dangers, especially cliffs and other drop-offs. We are also investigating tactile sensors to detect collisions with the body segments, especially the underside of Ratler (i.e., high-centering), and are extending the stereo-based obstacle avoidance planner to be more robust to sensor noise and uncertainty in the vehicle's interactions with the environment.

We also plan to add more extensive internal monitoring of the robot's health and integrity. For example, we will monitor battery voltage and motor currents, and autonomously "safe" the vehicle if they exceed given thresholds. We will also make use of various sensor redundancies to detect sensor failure itself. For instance, we can use the wheel encoders to estimate the change in orientation of the robot, which provides confidence that the turn-rate sensor is performing adequately.

In conclusion, we expect stereo perception, dead reckoning, and local obstacle avoidance to play essential roles in future lunar rovers. This work has demonstrated the successful reduction of these techniques from laboratory demonstration to reliable field practice. Further work and flight-qualification will permit these important capabilities to be put "on the shelf" for rovers exploring the Moon and beyond.

## Acknowledgments

This research was partly sponsored by NASA, under grants NAGW-3863 and NAGW-1175. We gratefully acknowledge assistance from Ben Brown, Michel Buffa, Lonnie Chrisman, Fabio Cozman, Yasutake Fuke, Richard Goodwin, Guillermo Heredia, Lalit Katragadda, Al Kelly, Paul Klarer, Sven Koenig, Luc Robert, Yoshikazu Shinoda, and Red Whittaker.

## References

[1] C. Angle and R. Brooks. Small Planetary Rovers. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pp. 383-388, Tsuchiura, Japan, July 1990.

[2] J. Bares and W. Whittaker, Configuration of Autonomous Walkers for Extreme Terrain, *Intl. J. Robotics Research*, 13(1):?-?, February 1994.

[3] L. Boissier, B. Hotz, C. Proy, O. Faugeras, P. Fua. Autonomous Planetary Rover (VAP): Onboard Perception System Concept and Stereovision by Correlation. In Proc. IEEE Intl. Conf. Robotics and Automation, pp. 181-186, Nice, France, May 1992.

[4] R. Chatila, R. Alami, S. Lacroix, J. Perret, and C. Proust. Planet Exploration by Robots: From Mission Planning to Autonomous Navigation. In P*roc. Intl. Conf. Advanced Robotics*, pp. 91-96, Tokyo, Japan, November 1993.

[5] F. Cozman and E. Krotkov. Digital Sextant. To appear in *Proc. IEEE Intl. Conf. Robotics and Automation*, Nagoya, Japan, May 1995.

[6] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press. 1994.

[7] L. Feng, J. Borenstein, and H. R. Everett. Where am I? Sensors and Methods for Autonomous Mobile Robot Positioning. Technical Report UM-MEAM-94-21, University of Michigan, December 1994.

[8] E. Gat, R. Desai, R. Ivlev, J. Loch, and D. Miller, Behavior Control for Robotic Exploration of Planetary Surfaces, *IEEE Transactions on Robotics and Automation*, 10(4):490-503, August 1994.

[9] K. Gatland. *Robot Explorers: The Encyclopedia of Spaceflight in Color*. Macmillan, New York, 1978.

[10] G. Giralt. Technical Thresholds between Robotic, Man-Tended and Permanently Manned Exploration and Exploitation of the Moon. In *Proc. ESA Intl. Lunar Workshop*, pp. 103-121, Beatenberg, Switzerland, June 1994.

[11] P. Grandjean and L. Matthies. Perception control for obstacle detection by a cross-country rover. In *Proc. IEEE Intl. Conf. Robotics and Automation*, May 1993.

[12] M. Hebert and E. Krotkov. 3-D Measurements from Imaging Laser Radars. *Intl. J. Image and Vision Computing*, 10(3):170-178, April 1992.

[13] M. Hebert and E. Krotkov. Local Perception for Mobile Robot Navigation in Natural Terrain: Two Approaches. In *Proc. Workshop on Computer Vision for Space Applications*, pp. 24-31, Antibes, France, September 1993.

[14] T. Kanade. Development of a Video-Rate Stereo Machine. In *Proc. ARPA Image Understanding Workshop*, Monterrey, California, November 1994.

[15] L. Katragadda, J. Murphy, and W. Whittaker. Rover Configuration for Entertainment-Based Lunar Excursion. In *International Lunar Exploration Conference*, San Diego, California, November 1994.

[16] A. Kelly. A Partial Analysis of the High Speed Autonomous Navigation Problem. Technical Report CMU-RI-TR-94-16, Robotics Institute, Carnegie Mellon University, 1994.

[17] A. Kelly. Adaptive Perception for Autonomous Vehicles. Technical Report CMU-RI-TR-94-18, Robotics Institute, Carnegie Mellon University, 1994.

[18] E. Krotkov, J. Bares, L. Katragadda, R. Simmons, and R. Whittaker. Lunar Rover Technology Demonstrations with Dante and Ratler. In *Proc. Intl. Symp. Artificial Intelligence, Robotics, and Automation for Space,* Jet Propulsion Laboratory, Pasadena, California, October 1994.

[19] E. Krotkov, M. Hebert, M. Buffa, F. Cozman, and L. Robert. Stereo Driving and Position Estimation for Autonomous Planetary Rovers. In *Proc. IARP Workshop on Robotics in Space*, Montreal, Canada, July 1994.

[20] E. Krotkov and R. Hoffman. Terrain Mapping for a Walking Planetary Rover. *IEEE Trans. Robotics and Automation*, 10(6):728-739, December 1994.

[21] E. Krotkov and R. Simmons, Perception, Planning, and Control for Autonomous Walking with the Ambler Planetary Rover. *Intl. J. Robotics Research*, To appear, 1995.

[22] P. Lescoe, D. Lavery, and R. Bedard. Navigation of military and space unmanned ground vehicles in unstructured terrains. In *Proc. Conf. Military Robotic Applications: Military Robotic Vehicles*, September 1991.

[23] L. Matthies. Stereo Vision for Planetary Rovers: Stochastic Modeling to Near Real-Time Implementation. *Intl. J. Computer Vision*, 8(1):71-91, July 1992.

[24] L. Matthies. Toward stochastic modeling of obstacle detectability in passive stereo range imagery. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, June 1992.

[25] L. Matthies and P. Grandjean. Stochastic performance modeling and evaluation of obstacle detectability with imaging sensors. *IEEE Trans. Robotics and Automation,* 10(6):783-791. December 1994.

[26] C. Proy, M. Lamboley, and L. Rastel. Autonomous navigation system for the Marsokhod rover project. In *Proc. Third Intl. Symposium on Artificial Inteligence, Robotics, and Automation for Space,* October 1994.

[27] C. Proy, B. Hotz, O. Faugeras, P. Gernesson, and M. Berthod. Onboard Vision System for a Mobile Planetary Exploration Robot. In *Proc. Workshop on Computer Vision for Space Applications*, pp. 2-12, Antibes, France, September 1993.

[28] J. Purvis and P. Klarer. RATLER: Robotic All Terrain Lunar Exploration Rover. In *Proc. Space Operations, Applications and Research Symposium*, Johnson Space Center, Houston Texas, 1992.

[29] L. Robert. Camera Calibration Without Feature Extraction. In *Proc. IEEE Intl. Conf. Pattern Recognition*, Jerusalem, 1994.

[30] B. Ross. A Practical Stereo Vision System. In *Proc. IEEE Intl. Conf. Computer Vision and Pattern Recognition*, pp. 148-153, New York. 1993.

[31] L. Robert, M. Buffa, and M. Hebert. Weakly-Calibrated Stereo Perception for Rover Navigation. In *Proc. ARPA Image Understanding Workshop*, pp. 1317-1324, Monterey, California, November 1994.

[32] R. Simmons. Structured Control for Autonomous Robots. *IEEE Transactions on Robotics and Automation*, 10(1):34-43, February 1994.

[33] H. Stone, Design and Control of the MESUR/Pathfinder Microrover. In *Proc. Intl. Conf. Advanced Robotics*, pp. 263-270, Tokyo, Japan, November 1993.

[34] C. Thorpe, editor. Vision and Navigation: the Carnegie Mellon Navlab. Kluwer, 1990.