Simplified Navigation and Traverse Planning for a Long-Range Planetary Rover

David P. Miller Li Tan Scott Swindell

dpmiller@ou.edu litan@ou.edu bacchus@ou.edu
Aerospace & Mechanical Engineering Computer Science
University of Oklahoma
Norman, OK 73019

Abstract

This paper presents the sensor and control system for SR2*, a planetary rover prototype capable of km+/day traverses through Mars-like terrain. The control system described allowed the rover to successfully navigate Mars-like terrain given a set of waypoints spaced approximately 100m apart. The control system uses input from a variety of sensors including heading, roll, pitch, an array of proximity sensors, voltage and current sensors, and a depth from stereo camera system. The control system is adequate, when combined with the rover's mobility system to make its way over or around almost all hazards a rover is likely to encounter, and to know when it is getting diverted so far off course that it needs to ask for help. In a recent field test, the robot autonomously navigated through 1.3 km of Mars like terrain to reach its goal. The SR2 is a solar powered, four-wheeled rover that masses 21kg and fits inside a* $90 \times 65 \times 40cm$ *envelope.*

1 Introduction & Motivation

The Sojourner (Mars Pathfinder) Rover was incredibly successful – but it only explored an area of Mars smaller than the front yard of a typical suburban track-house. The MER mission, scheduled to launch is 2003, using two much larger "long-range" rovers, will explore a section of Mars the size of a few football fields. However, if we are to get an understanding of Mars, or even of a particular area on Mars, we need to have the capability of performing truly long traverses to get to separated sampling sites, but maintain a sense of context on how they are connected [3, 11, 5].

Sojourner and the MER rovers are mechanically capable of traversing much larger distances than have been done or are being planned. But the control strategy of the rovers significantly reduces the distance the robots can travel. Effectively, the robots are never directed to travel over the horizon, or into an area that has not been carefully imaged. Due to rocks, uneven terrain, and limited camera resolution, seldom can a significant path be imaged sufficiently to meet NASA mission managers' constraints of acceptable foreknowledge.

The SR2 robot was created to gather experience and confidence that long traverses can be safely accomplished with a minimum of foreknowledge. SR2 has a number of sensors – many of which provide noisy, and sometimes unreliable data under field conditions. The terrain used in the field tests contained a number of hazards: steep slopes, ridges, rocks and holes in a distribution similar to that of the Pathfinder landing site [5]. Despite the unreliability of the sensors and the hazards of the terrain, SR2 was able to get to the assigned goals, progressing about 200m/hr towards its goal.

A key factor in the rover's capability to traverse more than a km/day towards its goal is the willingness of mission controllers to turn path planning over to the rover. As you will see, the rover is not particularly smart or efficient about its path, but it is unrelenting. Because there is no need to communicate back to Earth unless the rover gets *really* stuck[1], the SR2 rover is able to go tens to hundreds of times further in a day, through more hazardous terrain, then the Sojourner (and probably MER) rover(s).

The remainder of the paper will describe the navigation hardware and software for the robot. We will

[1] Really stuck means requiring human guidance to extricate, not irretrievably wedged. In the latter case, it is too late for communications.

touch upon some of the behaviors that have been implemented. We will conclude with a review of some of the results from a field test in the SaltonSea desert, and observation on the significance of these experiments.

2 Computational System Hardware

SR2 is designed to run in a desert environment using solar power. The onboard computer system is designed to handle and process stereo-vision data, sensor data, commands from the operator, control motors, and transfer images and other rover data between the rover and the ground station. A PC104 Plus system was chosen to fulfill these needs. The CPU board has a PIII 400MHz processor for all computation, one serial port to control two rover motors through a mini-SSC controller, one serial port to transfer compass data, and one USB port to communicate with the digital science cameras. A Wavelan wireless PC card is used for communication between the rover and ground station. In addition, an I/O board, IEEE1394 board and quad-decoder board are used for sensor, camera and motor encoder readings. All these boards and a power supply board are PC104 form factor. This power supply board also provides +5VDC for the USB hub, ranging sensors, encoders and other items.

3 Sensors

The task of the sensors in the SR2 is to detect obstacles, monitor the current and voltage of the solar panel and system, and to position the rover. A combination of a magnetic compass and drive motor encoders are used to position the rover. The encoders provide 64 counts per motor rotation or, after all of the gearing, 131 counts per cm traveled by the rover's wheels. The counts between the left and right hand side are averaged (if the rover spun in place by running the left and right wheels in opposite directions, then the average counts for that period should be 0). All heading information was derived by the compass [1]. The compass and encoders were updated at 15 Hz and the compass had an accuracy of $\pm 0.5^o$. The compass and encoder readings are combined to regularly update the robot's position.

The obstacles in the environment with which the rover must deal include big rocks, holes, and slopes over which the rover cannot climb. The characteristics of the desert environment must be considered when selecting sensors. The biggest problem we faced was high contrast and unusually bright lighting (especially in the near IR).

SR2 was initially assembled using the Sharp GP2D12 infrared distance measuring sensor and Videre Design's stereo DCAM head. We used SVS (Small Vision System) [4] as our stereovision process API. However these did not work well in the extreme lighting conditions of the field test. Under these lighting conditions, the cameras lacked the dynamic range needed to produce images that could be correctly processed by the SVS. As a result, stereo was not used for navigation during the primary field tests.

After extensive experimentation, we found that Sharp range sensors such as the GP2D12 [13] would not give accurate range data under the bright desert lighting conditions. However, they did prove to be reliable as threshold detectors under those conditions. The threshold range was extended by switching to a new lens package for the detector - the Sharp GP2Y0A02YK which provides longer range optics and better sun shielding. Using the sensors in a threshold mode to determine whether or not there was an obstacle in the robot's way, or whether or not the ground had fallen away proved to be much more noise tolerant.

Current sensors and voltage dividers were used to monitor system current and solar panel voltages. The system could switch to SLEEP mode if the battery voltage was too low and wake up when the battery is charged, though this capability was never tested in the field. The wireless communications card could be switched to SLEEP mode in order to save power, but its resumption had to be done by a timer.

The sample rate of the I/O board also affects the sensor reading. If the rate is too fast, the system could not get stable readings, while the system could not respond in time if the sampling is too slow [10]. The sampling rate of Sharp sensors was set to 15Hz after all factors were taken into account.

The compass has direct heading, pitch and roll serial port outputs. The pitch and roll are used to determine if the slope is too high or the rover is rolling too much.

Motor current was used to detect if a motor stalled and also as part of the power monitoring system. The power used by the sensors and computation system were also monitored.

The SR2 computer is a 400MHz PIII which runs Red Hat Linux 7.3. The processor speed is adequate to process a stereo pair a few times a minute. When not running stereo, the processor is idle most of the

time.

4 Path Planning

The flow of the low-level control is shown in Figure 1. When the stereo vision is operational, SR2 uses a variation on the NaT algorithm [9, 12]. This algorithm calculates the instantaneous heading and speed for the rover from the composition of spun potential fields that are placed in the map representation for each physical obstacle. The spinning of the fields assures that the vehicle will not get stuck in a local minima. The vision system is used to generate the NaT map. The other sensors can cause the robot to stop or take immediate avoidance action if the robot gets too close to an obstacle.

The SR2 server is the interface between the low-level navigation system (written in C++) and the JAVA rover controller, which incorporates movement behaviors, and talks to the mission planner.

5 Onboard Robot Control

SR2 can either be tele-operated or put in autonomous mode. In autonomous mode, it controls itself and avoids obstacles in order to reach a goal, or series of goals, specified by the user.

5.1 Autonomous navigation

For autonomous navigation, the user must first specify a mission. A mission consists of a final destination, called the goal, and optionally, a series of intermediate points that it should pass through to reach the goal, called waypoints. The waypoints allow the user to set a path around large known obstacles in order to reduce the time required for the robot to reach the goal. Both waypoints and goal points are specified in rectangular coordinates, as is the current location of the rover (a polar coordinate input option is also available to the user).

In order to be able to travel to a given point, the rover must first know its current location. When starting a mission, the user will typically reset the rovers location to the origin to make specifying waypoints and goal points simpler. From there, its up to the rover to track its own position as it carries out the mission (see section 3).

5.2 Obstacle Detection and Avoidance

When in autonomous navigation mode, the rover tries to travel in a straight line to the next waypoint or goal, but obstacles may lie in the chosen path. Not counting the stereo vision system, there are two methods the rover uses to detect obstacles: range sensors and electronic compass data.

The rover's ten forward facing range sensors are clustered into three logical categories: right, middle, and left. When an obstacle is detected by the range sensors, the exact behavior chosen for avoidance varies based on the cluster that detected the obstacle. If the right cluster detects an obstacle, the rover backs up for several seconds, then rotates briefly to the left (counter-clockwise) and attempts to travel forward for several seconds. This procedure is repeated until the right sensor cluster no longer detects any obstacles and the rover is able to travel forward without interruption for a specified number of seconds[6]. Once the rover has successfully gone around the obstacle, it recalculates the path to the next destination based upon its new location [7]. The same algorithm is used to avoid obstacles detected by the left sensor cluster, with the exception of rotating to the right (clockwise) rather than the left. Detected holes or cliffs are treated the same as obstacles.

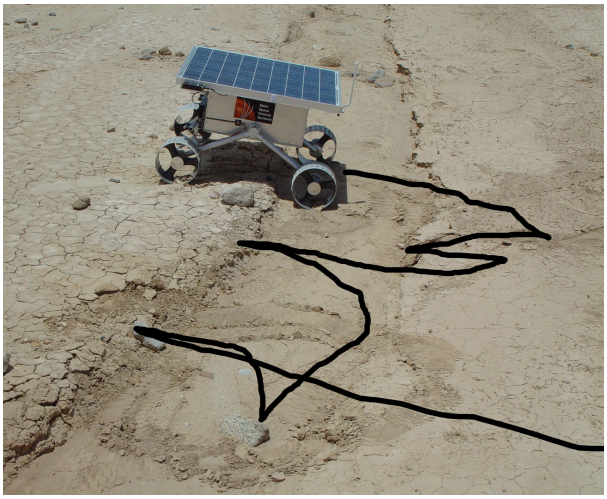

Figure 2: *SR2 moving along a ridge-line until an acceptable slope is found (pathline added)*

The middle sensor cluster is set back farther than the left and right sensors. As a result, obstacles detected by the middle cluster are typically detected when they are at a much closer distance than those detected by the left or right sensors. Additionally,

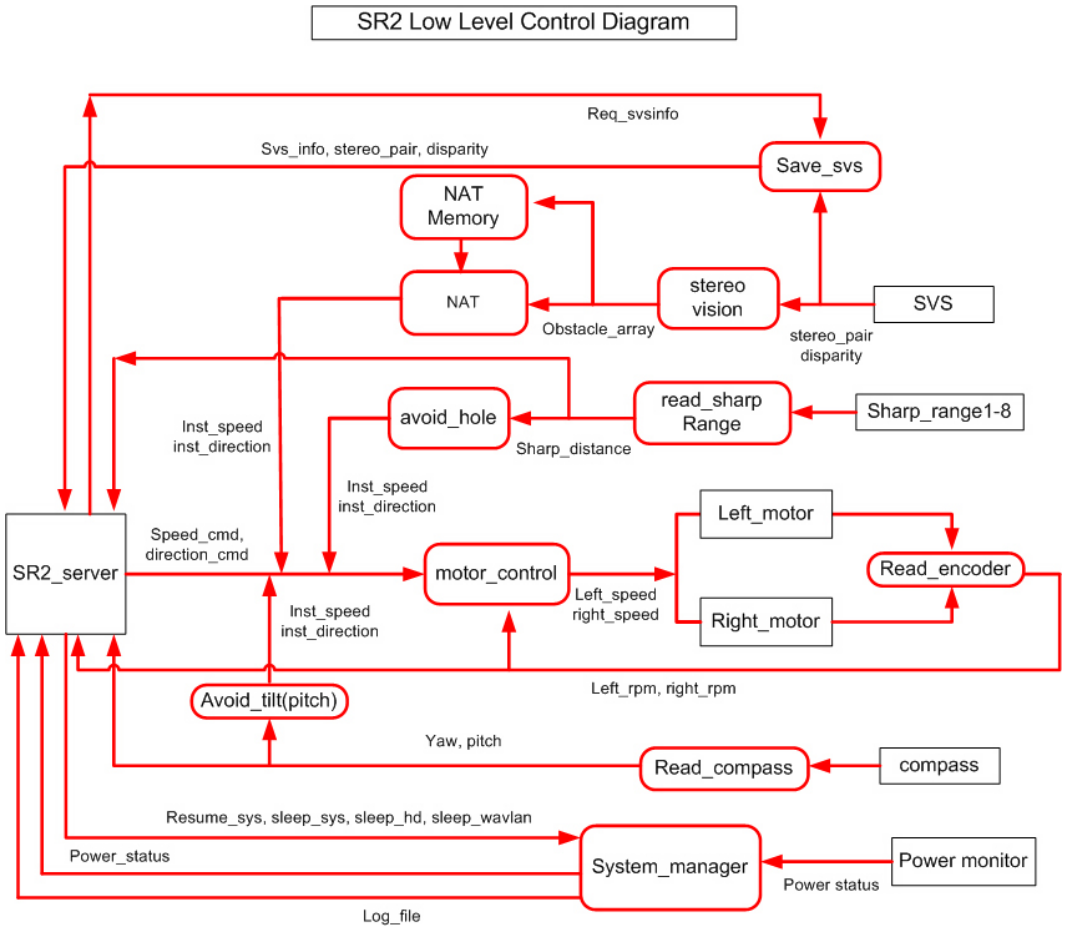

Figure 1: *Flow of Control in SR2*

since this cluster detects obstacles that are directly in front of the rover, a larger correction is needed to go around the obstacles and so the rover turns for a longer period. Aside from that, the algorithm for avoiding obstacles in the middle of the range of view was nearly identical to that for the left and right sensors. The middle cluster was actually subdivided into left and right sensors to allow easy determination of which direction to rotate when avoiding an obstacle. In addition to range sensors, the rover would also take into account pitch and roll readings to avoid trying to navigate up or down too extreme of a slope.

During field trials, this technique of obstacle avoidance proved quite effective after the sensors were properly calibrated and ideal durations of each movement phase (reverse, rotate, forward) were determined. The rover was even able to navigate out of semi-circular arrangements of rocks. By continually rotating in the same direction until no obstacle is detected it ensures, that at worst, the rover would have to backtrack the way it came to avoid complex obstacles, such as those that completely surrounded it (e.g., Figure 2).

6 Remote User Interface

The operator of the rover issues commands and receives feedback via a remote networked graphical interface. All Commands are human readable text and can be entered on a simple command line if desired, or more easily via graphical controls. Most of the sensor and status feedback sent from the rover is also in the form of human readable text with the exception of images, obstacle data, and mission path data. The graphical display and controls were divided into two separate panels: engineering and mission control.

6.1 Engineering Panel

The engineering panel displays real-time log file updates from the rover. In addition, it contains the simple command line for manually issuing text commands to the rover. It also contains a simple array of nine buttons to manually control the movement of the rover in any direction, as well manual controls for each motor. In addition to the real-time log file display, this panel also displays the encoder readings of the motors to give a rough estimate of speed, as well as power meters for the computer, batteries, and solar panel. The engineering panel is primarily intended to testing purposes as well as monitoring low-level system functions.

6.2 Mission Panel

The mission control panel offers more high-level functionality than the engineering panel. The mission panel contains a graphical compass to show the rovers current heading as well as pitch and roll readings. In addition to the compass display, the mission panel also graphically displays sensor readings arranged around a small icon of the rover to make it easier for the remote user to interpret the readings (see Figure 3.

The center of the mission panel is dedicated to displaying images from the science cameras. These are special cameras mounted on a structure on top of the rover which could be commanded to take stereo images with the click of a button. The rover would then upload the images to the remote interface for displaying. In addition to the science camera images, the current images being analyzed by the navigational cameras could also be retrieved and displayed in a separate dialog with the option of automatic updating.

The primary method of control available on this panel was the mission planner, which provided a graphical way of setting way-points and goals. Once the user was satisfied with the mission it could then be sent to the rover to be carried out under autonomous navigation. During the actual execution of the mission, the mission planner would update the location of the rover icon to show the current location of the rover and the actual path taken between waypoints.

7 Tests & Results

In June of 2002 the SR2 underwent tests in the Saltonsea desert in California. Prior to the main run, a closed loop course at the desert field test site was used to determine dead-reckoning errors. Several runs of 200m or more were done and found that the cumulative dead reckoning position error yielded a finish position within 2% of distance traveled from the start position, even through rugged terrain with slopes of up to 15^o. The precise heading information from the compass (or heading gyro in a space mission) makes dead reckoning quite reliable. The SR2 wheels and tread were specifically designed to minimize slippage. The remaining slippage was random and mostly cancelled out leaving only a small resultant error.

On the final day of testing, the rover was given a series of way points spaced an average of 122m apart with the closest pair about 25m apart and the farthest just under 200m. The total length of the traverse was 1.3km. The way-points were selected in order to steer the robot around major geologic features. These features were all of sufficient size that they would have been easily detectable from an orbiting camera of the kind planned for future Mars missions.

The way-points were uploaded to the rover, and it started its traverse at 12:30pm, During the next five hours, the robot autonomously traveled between way points. Direct intervention was needed to restore the robot's state after a shutdown caused by a heat related mechanical failure (see [8]), and for configuring the robot to take "science" images. The robot operated completely autonomously for navigating between waypoints and for obstacle avoidance.

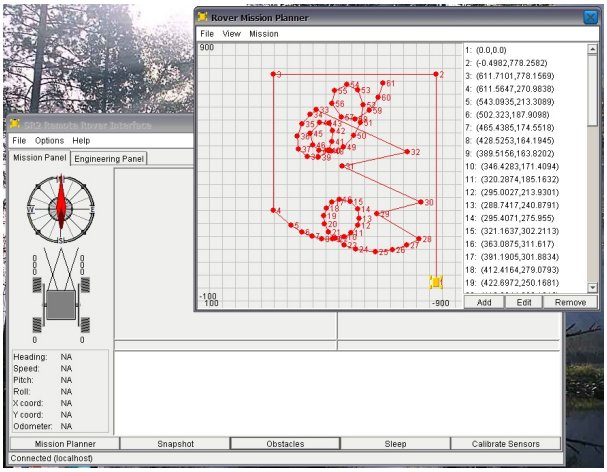

Figure 3: *The Mission Panel*

The exact path chosen by the rover occasionally caused some temporary distress to those monitoring the field test. In order to keep from getting stuck against a rock, or digging up to much soil during skid turns, the rover was programmed to rock forwards and back while performing skid steers. This worked well

but took time, so the robot only turned when needed to go around an obstacle, or when it's heading deviated from the goal by more than 10^o. When the robot came upon a large rock or ridge-line it seemed it more often than not would turn the "wrong" way, resulting in more avoidance maneuvers than would have been necessary if the rover had been "smarter".

Another behavior of SR2's was the maximum slope avoidance. If a slope exceeded 15^o, the robot would alter its heading until an acceptable slope angle was reached. After traveling a couple meters, the rover would turn back towards its next way-point. During this correction, the slope would often increase causing the robot to again turn away. Depending on where in the corrective turn the slope exception was discovered, the rover would either tack back and forth like a sail boat going into the wind, or would describe a scalloped shape course. To the uninitiated viewer, the robot's actions often appeared horribly inefficient.

However, all of these observational anxieties were groundless and could be relieved by simply *not* following the robot continuously, but rather checking in with it every fifteen minutes or so. At fifteen minute or larger intervals, the robot had always made significant progress, and had always managed to *unstick* itself.

While not burdened by the science requirements of Sojourner, SR2 was able to navigate further in 30 minutes than Sojourner did in 80 days through similar terrain and with a vehicle of the same general size. We believe that SR2 is the first vehicle of this class to accomplish this level of autonomously navigated and powered traverse. A key result is that the rover was able to navigate around hundreds of obstacles and find the pathways through ridge lines without any long range sensors. The stereo vision was not used. The range sensors were used as threshold proximity sensors. Simple sensing combined with a few clever, but simple behaviors is very powerful. This has been established in the literature for short range indoor robots [2]. We believe that SR2 shows that this strategy is a suitable replacement or at least supplement to current mission philosophy. We will not be able to remotely explore Mars until we are willing to delegate some level of control to the robots.

Acknowledgments

The authors wish to thank the other members of the SR2 engineering team: Matt Roman, Tim Hunt & Alois Winterholler and the SR2 science team Mike Malin & Mike Ravine. This work was funded in part by a grant from Malin Space Science Systems Inc.

References

[1] M. Bohlinger. Magnetic sensor description hmr3000. http://www.ssec.honeywell.com-/magnetic/description/desc_3000.html, 2002.

[2] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation*, RA-2(1), Mar 1986.

[3] N. Cabrol, E. Grin, V. Gulick, C. McKay, R. Greeley, M. Sims, and G. Briggs. Rover mobility & sampling strategy on mars: The case for gusev crater. In *Proceedings of the Lunar Planetary Science Conference #27*, 1996.

[4] K. Konolige. http://www.ai.sri.com/software-/SVS, 2002.

[5] M. Malin. Personal communication., April 2002.

[6] D. P. Miller. Rover navigation through behavior modification. In *Proceedings of the NASA Space Operations Automation and Robotics Workshop*, Albuquerque, NM, June 1990.

[7] D. P. Miller, R. S. Desai, E. Gat, R. Ivlev, and J. Loch. Reactive navigation through rough terrain: Experimental results. In *Proceedings of the 1992 National Conference on Artificial Intelligence*, pages 823–828, San Jose, CA, 1992.

[8] D. P. Miller, T. Hunt, M. Roman, S. Swindell, L. Tan, and A. Winterholler. Experiments with a long-range planetary rover. In *Proceedings of the The 7th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, Nara, Japan, May 2003. ISAS, NAL, NASDA.

[9] Marc Slack. *Situationally Driven Local Navigation for Mobile Robots*. PhD thesis, Department of Computer Science, Virginia Tech., 1990.

[10] D. B. Stewart. How to choose a sensible sampling rate. *Embedded System*, pages 20–27, July 2002.

[11] C. Stoker. The search for life on mars: The role of rovers. *Journal of Geophysical Research*, 103(E12):28557–28575, 1998.

[12] L. Tan and D. P. Miller. Navigation templates for psa. In *Proceedings of the First International NAISO Conference on Autonomous Intelligent Systems*, Geelong, Australia, 2002.

[13] J. C. Zufferey. Sharp gp2d12. http://dmtwww.epfl.ch/j̃zuffere-/SharpGP2D12_E.html, April 2001.