Reactive Speed Control System Based on Terrain Roughness Detection

Mattia Castelnovi
Laboratorium DIST
University of Genova
mattia.castelnovi@dist.unige.it

Ronald Arkin
College of Computing
Georgia Institute of Technology
arkin@cc.gatech.edu

Thomas R. Collins
School of ECE / GTRI
Georgia Institute of Technology
tom.collins@ieee.org

***Abstract* – Autonomous outdoor navigation requires the ability to discriminate among different types of terrain. A non-trivial problem is to manage the robot's speed based on terrain roughness. This paper presents a speed control system for a robotic platform traveling over natural terrain. This system is based on the view of a line-scanning laser of the area just in front of the platform. Analysis of range data for roughness produced by the laser over different terrains is examined. An algorithm for managing speed through different terrain has been tested on real outdoor surfaces producing excellent performance.**

***Index Terms* – reactive control, off-road navigation, terrain analysis, surface roughness.**

I. Introduction And Related Work

Autonomous and semi-autonomous cross-country navigation has been of interest over the past two decades because autonomous vehicles hold great promise for applications in many domains such as agriculture [1], mining, space exploration [2, 3, 4], healthcare and service robotics [17], and military/security missions [5]. One of the biggest challenges in this field is environmental perception, with terrain feature detection being particularly difficult [6, 7]. Usually, robotic off-road systems must deal with two fundamental problems: the imprecision of sensor measurements that characterize the perception of natural environments and the online, real-time acquisition and analysis of terrain surface data [18]. Because of these problems, outdoor robot navigation is usually more challenging than indoor robot navigation. Although most of the literature pertaining to this topic is dedicated to off-road obstacle detection and/or obstacle avoidance [8, 9, 10], the study of terrain surface roughness detection as an obstacle is an important step toward real autonomous navigation through unknown terrain [1, 11, 12].

Terrain geometry is critical in mobile outdoor robotics since uneven areas require consideration of how the vehicle will interact with upcoming terrain in order to keep the vehicle safe. Obstacle avoidance and terrain surface roughness detection are tightly linked since, on rough terrain, the notion of an "obstacle" depends on the capacity of the robotic system to overcome terrain irregularities. Although research in off-road navigation of robotic platforms through unknown terrain has progressed significantly, safe and robust techniques for limited computing and sensing resources have not yet been produced [8].

This paper focuses on the problem of extracting ground terrain surface geometry measurements in order to manage the speed of the robot for safe teleoperation. Regardless of the surface type, the commanded speed of the platform has to consider the roughness of the surface in order to navigate safely. A central goal for our system is to enable untrained people to directly control a robotic platform while maintaining the integrity of the robot itself and the mission. The next section describes the robotic platform that is currently being used for our experiment. Section III is focused on the system procedure for terrain roughness detection. Section IV discusses the algorithm used to manage the speed of the robot. Section V describes experimental results. The final section is devoted to conclusions.

II. Robot Platform

A. Hardware system

Our system is a Segway RMP (Fig. 1), a new mobile platform based on the Segway Human Transporter (HT).

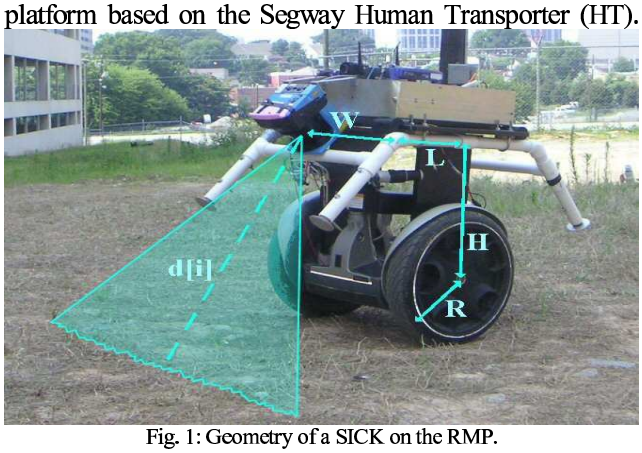


Fig. 1: Geometry of a SICK on the RMP.

The RMP can operate outdoors at speeds up to 3.5 m/s, can sustain a maximum acceleration of 2 m/s^2, has a range of 15 km, and can carry up to 50 kg [16]. The RMP tilts forward when accelerating and backward when decelerating; its pitch angle can easily reach 40 degrees in either direction depending on the acceleration input. The RMP updates (e.g., attitude, velocity) are acquired at about 50 Hz via a CAN interface. For our purposes, the RMP is equipped with a SICK laser range finder, mounted with the scanning plane angled downward in the direction of motion. The laser plane projects towards the ground at approximately 45 degrees. This configuration of the laser [13] provides a clear view from above, but affords only a

very short time to react to changes in terrain roughness. The laser is configured to scan a 180° radial field with a range of 0.5°. Scans are generated at 4 Hz, which could be increased to about 40 Hz at the expense of increased computational load, an issue for the objective of fast reactive control. To collect data from the laser scanner and to manage commands, an on-board "host" laptop (Dell Latitude with Pentium II 233MHz, 512K Cache, 64MB of memory) is mounted in a ruggedized case on the top of the RMP. An 802.11 Linksys wireless access point is connected with the laptop to allow the robot to communicate with an operator control unit (OCU). The OCU is an Acer Aspire Pentium 4 2.6 GHz laptop connected via an Orinoco 802.11 PCMCIA card. The OCU's purpose is to allow an operator to drive the RMP with a Logitech Wingman joystick.

B. Software System

Linux (RedHat 9.1) is installed on both laptops. The RMP drivers run as part of HServer, an application which provides a uniform device interface across various robot platforms and sensors for the *MissionLab* system. *MissionLab* v6.0 [14, 15] is the multiagent robotic mission specification and control system used to acquire the data and to control the robot. *MissionLab* generates the executable code specific to the RMP and allows shared usage of other drivers (e.g., the laser scan interface) across a variety of robotic platforms. The experimental results are hence general and transformable to any other supported robotic platform (although the RMP itself presents more control problems than most mobile robots).

III. PROCEDURE FOR TERRAIN DETECTION

This section explains how the system is able to process data to choose the right speed for each type of terrain, using the array of distances provided by the laser scanner.

A. Offline Settings

Because it is impossible to be certain about the orientation of the scanning plane, it could be hypothesized that the laser scanner has some bias both in the roll angle and in the yaw angle. Calibration helps the system to take more objective data. Moreover, since the system is going to detect roughness, it is very important to start with a good measure of flatness in order to compare the values it is going to calculate. For these reasons, before starting to compute the roughness of terrain, it is necessary to take a scan of the height of flat terrain in front of the robot and store in an array called $height_{flatterrain}[i]$. This has to be done just once at the beginning, before each data run. Online, the system subtracts $height_{flatterrain}[i]$ from each value calculated in the i-th position (Fig. 3).

B. Online Procedure for Roughness Measurements

Fig. 3 shows the overall procedure of the detection system, which is the first processing stage. The following steps are executed in order to obtain useful terrain roughness parameters.

1. *Calculate height.* Scans are transformed into position and height values. Given the *robot tilt angle* θ, the *radius* of the wheels R (see Fig. 1 and Fig. 2), the *height of the laser scanner* from the wheel center H, the *horizontal distance* from the wheel center to the laser scanner L, the *height* of a certain terrain point detected by the i-th laser beam is:

$$height[i] = R + H\cos\theta - L\sin\theta - d[i]\cos(\alpha - \theta) \quad (1)$$

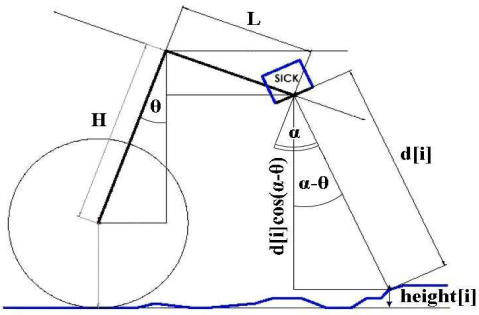


Fig. 2: Angles used in the algorithm.

where d[i] $\cos(\alpha - \theta)$ is the vertical factor of the i-th laser reading (Fig. 1 and Fig. 2) and α is the *downward pointing angle* of the laser scanner (≈45 degs). Note that d[i] is not the raw laser range reading, but rather the sensed distance *perpendicular to the face* of the laser scanner (i.e., only in the case of the centermost laser reading is d[i] equal to the raw reading.

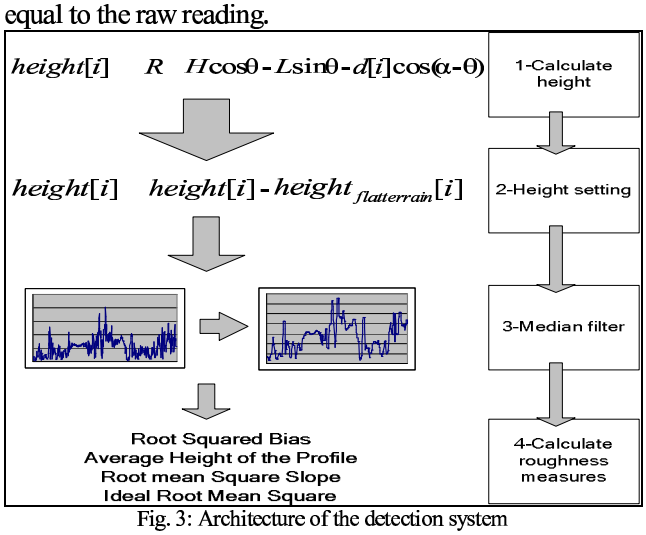


Fig. 3: Architecture of the detection system

2. *Height setting.* For each value the bias calculated offline is subtracted:

$$height[i] = height[i] - height_{flatterrain}[i] \quad (2)$$

3. *Median filter.* Since laser readings are very noisy, (according to the technical information of the scanner, the statistical error in the range from 1m to 8m is about ±15mm) during robot motion a median filter windowed on the last n readings is applied in order to smooth the data. Here, n=7.
4. *Calculate roughness measures.* Finally, surface roughness parameters are calculated in order to evaluate the terrain that the system is approaching. All computed values are based on a single laser-scan reading. After several experiments on different terrain, the most useful roughness parameters for our purpose have been selected:

- **Root Squared Bias (*RSB*) average roughness** – the variance of the amplitude distribution function of the height of the terrain profile (3),
- **Average Absolute Slope (*AAS*)** – the average of the absolute slope value between scan points perpendicular to the direction of motion (4),
- **Root-Mean-Square Average Slope (*RMSAS*)** – the RMS value of the same slope used in the AAS metric (5), and
- **Ideal Root Mean Square (*IRMS*)** – the RMS calculated with a ideal flat surface average value of zero (6).

Each roughness parameter ranges from zero for flat surfaces to a maximum value experimentally detected.

$$RSB = \sqrt{\frac{1}{120}\sum_{i=0}^{119}(height[i] - av_{lastline})^2} \quad (3)$$

$$AAS = \frac{1}{120}\sum_{i=1}^{120}|height[i-1] - height[i]| \quad (4)$$

$$RMSAS = \sqrt{\frac{1}{120}\sum_{i=1}^{120}(height[i-1] - height[i])^2} \quad (5)$$

$$IRMS = \sqrt{\frac{1}{120}\sum_{i=0}^{119} height[i]^2} \quad (6)$$

The *RSB* value (3) is similar to a root-mean-square calculation, but the average height of the previous scan is used instead of the average in the current scan, so the subtracted bias is not the actual mean. Using the previous average, measured in the same general terrain, results in a sensitivity to sudden discontinuities *in the direction of motion* that is captured by none of the other three roughness parameters.

IV. Speed Control

As noted, the basic objective is not a complete high-level control system, but a low-level reactive speed controller. Its goal is to allow the user to direct the movement of the robot, with the robot taking initiative concerning its own speed in order to protect itself from damage. Specifically, a user will be able to request a velocity that is too high (by driving a joystick to its limit), but the control system should safely govern the commanded speed. This goal might seem easily obtainable using thresholding. For example, one could choose *n* different speed values, associate them with *n* different regions in roughness parameter measurements, and govern the speed at *n* different levels.

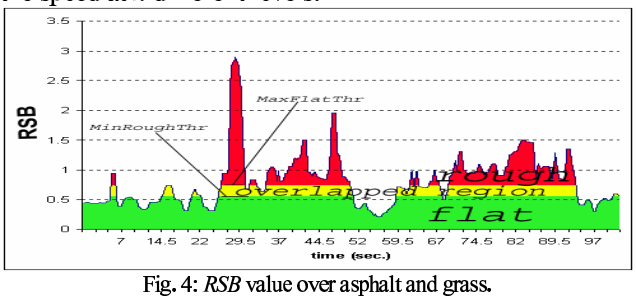


Fig. 4: *RSB* value over asphalt and grass.

Unfortunately, in the desired application (off-road navigation with a platform that frequently tilts to view different scan lines of terrain), measurements vary so drastically that it is difficult to recognize clearly the differences between safe terrain and hazardous terrain. Fig. 4 shows the values of the *RSB* parameter as a function of time during an experimental run over asphalt and grass. The plot illustrates the difficulties in detectionThe green region (lower part of the graph) designates the range that experiments showed to usually represent flat terrain, while the red region (higher, darkest part of the graph) designates the range that proved to usually represent rough terrain. In the yellow region (middle, lightest part of the graph), it is hard to decide . if we are driving on flat terrain or on a rough surface. To solve this issue, hysteresis was introduced. Two different thresholds (*MaxFlatThr* and *MinRoughThr*, with *MaxFlatThr*>*MinRoughThr*) are fixed for each of the four different roughness parameters (*RSB, AAS, RMSAS, IRMS*). *MaxFlatThr* is the maximum value attributed to flat surfaces (upper limit of yellow area), and *MinRoughThr* is the minimum value assigned to rough/hazardous surfaces (lower limit of yellow area). As depicted in Fig. 4, the region considered as flat (*RSB*< *MaxFlatThr*) overlaps part of the region considered as rough (for *RMS*>*MinRoughThr*).

```
SpeedControlAlgorithm
 1 for parameter ← (RMS, AAS, RMSAS, IRMS)
 2   do MaxFlatThr_parameter ← max_parameter for flat terrain
 3      MinRoughThr_parameter ← min_parameter for rough terrain
 4      if parameter > MinRoughThr_parameter
 5         then RoughRank++
 6      if parameter < MaxFlatThr_parameter
 7         then FlatRank++
 8 FinalRank = FlatRank- RoughRank
 9 if FinalRank >0
10    then SpeedValue++
11 if FinalRank ≤ 0
12    then SpeedValue--
13 if FlatRank == 4
14    then SpeedValue = top-speed
15 if RoughRank == 4
16    then SpeedValue = safe-speed
```

Fig. 5: Speed Control Algorithm pseudo-code

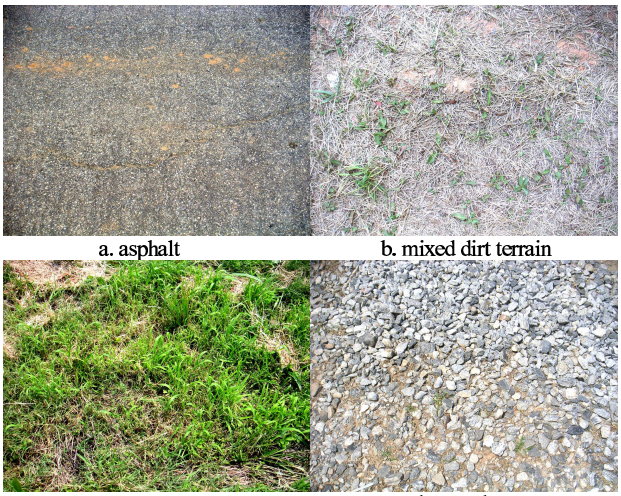
The hysteresis associated with the overlapping region serves as a guard against frequent speed transitions. The algorithm is rate-limited by the laser scan rate. To be as fast as possible, the control system constantly (4 Hz) checks the four different roughness values. A counter, *FlatRank,* is increased if the current value is below the *MaxFlatThr* (line 4 and 5 in Fig. 5) and another counter, *RoughRank,* is increased if the current value is higher than *MinRoughThr* (line 6 and 7 in Fig. 5). *FinalRank* – defined as the difference of *FlatRank* and *RoughRank* – is calculated in order to evaluate the roughness of the terrain the robot is approaching (line 8 in Fig. 5). To better manage the speed of the robotic platform, four different commanded values are the *top-speed*, *mid-high-speed, mid-low-speed* and *safety-speed*.

The two mid-speed values are used for two different reasons. First, it helps to generate smoother motion for better acquisition of data from the sensors (especially with

platforms like the RMP that tilt). The second reason is that, with two more values, the system is better able to move along in different types of terrain with different roughness values. Four times per second, the system checks and manages the speed. It increases the speed by one step (from safety-speed to mid-low speed e.g.) if the *FinalRank* is positive (lines 9 and 10 in Fig. 5), it decreases the speed by 1 step if *FinalRank* is negative or zero (lines 11 and 12 in Fig. 5). Lines 15 and 16 describe the behavior of the system when all roughness parameters are over *MinRoughThr:* the speed is set to safety-speed regardless of the value of *FinalRank*. This restriction assures that the command for the slowest speed is launched in the shortest time possible (0.25 seconds) when very rough terrain is detected. (With the chosen hardware, the computational time is very small compared to the laser scan acquisition, so the scan period of about 0.25 seconds dominates the control cycle.) Throttling the speed in one scan interval could be essential in preserving the safety of the platform. In the same way, when *FlatRank* is equal to 4, the speed is set to top speed (unless *RoughRank* is also 4) in order to have a constant speed on flat terrain even if some roughness parameters are in the overlapped region.

V. Experimental Results

The system has been tested in different terrain transitions for more than seven hours in outdoor environments. In all of those experiments, four different kinds of terrain were considered: asphalt, mixed dirt, grass, and gravel (Fig. 6).



a. asphalt b. mixed dirt terrain

c. grass d. gravel

Fig. 6: Tested terrain

A. Gravel-Asphalt Experiment

This experiment involved an RMP run over asphalt and gravel alternately, as shown in Fig. 7. The user is pushing the joystick as hard as he can for the duration, except when turning. It has been found that it is more challenging to detect gravel than grass because the difference of height detected with laser scanner between the peaks and valley in gravel is smaller than in terrain like grass. Analyzing data acquired for this experiment, it is possible to better understand how the system works. Figures 8-10 show speed, tilt angle, and parameter measurements for the asphalt/gravel experiment. Initially, calibration is performed manually as it is depicted in the highlighted region in Fig. 10. After calibrating, it is possible to start safely driving the RMP (the 8-second mark, noted both in Fig. 7 and in Fig. 8). During this period, all roughness parameters increase, even if the terrain in front of the RMP remains the same. That happens because the RMP's pitch angle increases most at the time at which the RMP begins to accelerate. After reaching the commanded speed, the pitch angle decreases to find the steady-state balancing point at that velocity (Fig. 9), although minor corrections continue to be made at a high frequency which is difficult to observe in this data.

Fig. 7: Path of the gravel-asphalt experiment.

Between 15 and 23 seconds, the robot turns to point at the gravel and then approaches it (after 23 seconds). It is interesting to see how the *RSB* value seems to be most correct in this situation; its values are all over the *MaxFlatThr* threshold, while *AAS, RMSAS* and *IRMS* continue to stay under the roughness threshold.

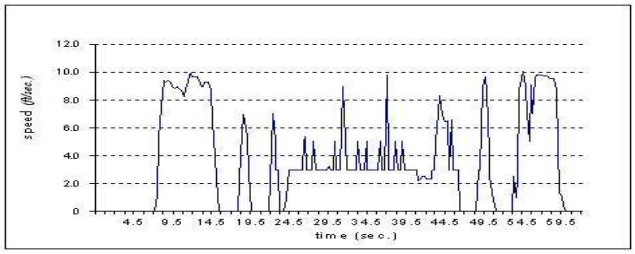


Fig. 8: Actual speed plot of the gravel-asphalt experiment

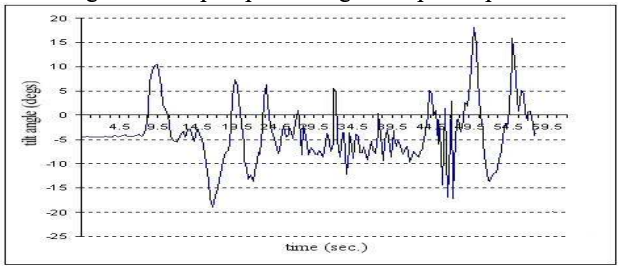


Fig. 9: Tilt angle plot of the gravel-asphalt experiment.

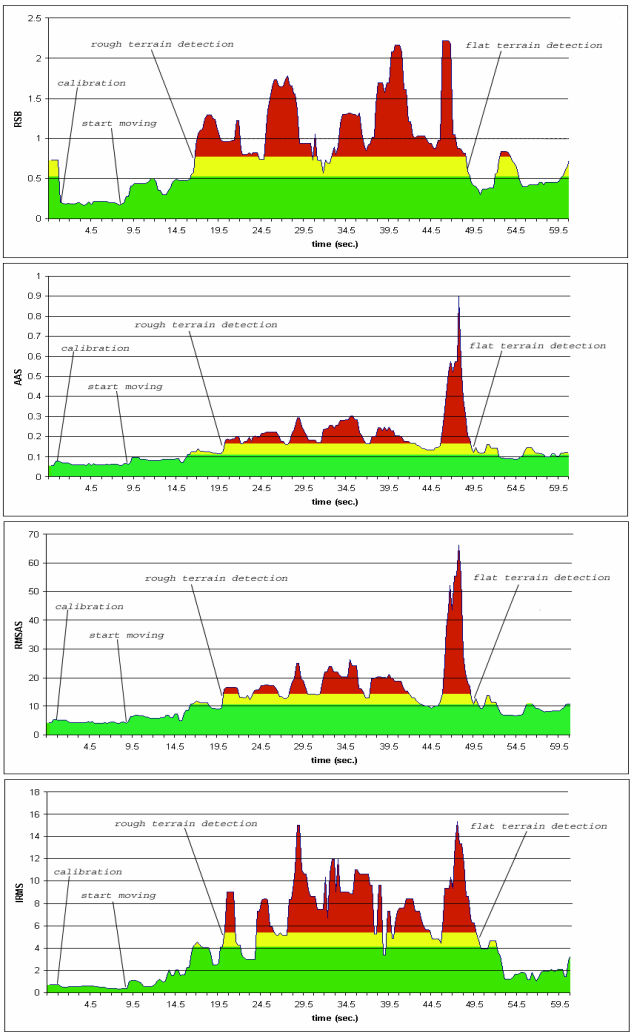


Fig. 10: Roughness parameter plots of the gravel-asphalt experiment.

The system begins to perceive a more rugged surface, even though most of the measures suggest the terrain is not rough enough to require the safety-speed. At this point, we have one parameter (*RSB)* voting for uneven terrain and three parameters located in the yellow (overlapped) zone. This means that the system is not able to confidently state the ranking of roughness terrain, but it is surely not so hazardous as to require a speed below the safety-speed. In this case, the algorithm determines that *RoughRank* is four and that *FlatRank* is 3. Since *FinalRank* (line 8 Fig. 5) is equal to -1, the speed-threshold is decreased by one step (from top-speed to mid-high speed in this case). On the upper-right corner of the path (Fig. 7), the laser is pointing ahead towards the dirt terrain at the right. By turning again it is possible to detect gravel, as shown in the plots after 20 seconds. From 20 to 47 seconds, most of the time the RMP is not allowed to exceed the safety-speed regardless of the speed commanded by the driver because all the roughness parameters are over *FlatThr*.

The speed plot shows that occasionally the speed exceeds the safety-speed threshold. This is due to some lacks of gravel on the ground, where dirt soil appears flat. These detections are not to be considered as errors because the terrain is really flat when gravel is not enough to cover the dirt terrain. However, in this case, the shifting of the speed threshold does not compromise the safety of the robot because their duration is only 0.25 seconds long (just one scan). The RMP does not have time to increase its velocity significantly. In spite of the speed plot appearance, the produced motion is very smooth. After ≈54 seconds, the RMP turns into the road to close the loop and suddenly starts to increase its speed until a command to stop is sent (after ≈61 seconds).

B. Asphalt-Grass Experiment

Fig. 11 shows plots of data from a path over grass and asphalt. In this case the robot was never stopped to make it turn because more space with the same kind of terrain was available. It is important to note that the speed-control plot shows the values of speed-thresholds as a function of time: value = 1 means safety-speed is the maximum speed allowed, value = 2 mid-low speed is allowed, and so on. The driver, even an inexperienced one, can drive as fast as he likes but cannot exceed the speed threshold imposed by the system.

In this experiment, it should be clear that the four roughness parameters used are able to complement each other. In fact, the *IRMS* parameter, which was very accurate in the previous experiment, is the one that, in the present experiment, decreases too early, failing to detect the grass (after ≈79 seconds). The other three values are able to prevent the robot from going faster, as is correct when driving over grass.

VI. CONCLUSIONS

This paper presents a speed control system for safeguarded teleoperation that can be extended to support autonomous behaviors. The results from experiments conducted on some common off-road surfaces have been shown. The developed system safely operates a relatively dangerous, fast and heavy robotic platform for its size. Before and during the system's development, many runs were made over the terrains analyzed in this paper. Without the speed control system, the RMP fell every time it was run over rough terrain if the top speed was commanded. Although the user was always trying to drive it safely, the RMP broke its protective arms over 10 times in more than 100 runs in outdoor environments. After developing the speed control system, the RMP fell three times in seven hours of testing. One instance was due to a low battery charge, while the other two instances were due to terrain transitions like a step or a hole. The step between asphalt and grass, for example, sometimes is followed by irregularly sloped terrain, which requires a very fast response time to be detected and to take the right action. A report by Bau [19] shows that RMP has a lag in response time which lasts approximately 0.4 seconds and it is not related to magnitude of velocity command. This handicap is not due to the system presented, but it is something specific to the RMP platform. (At top speed, the RMP is able to cross 1.4 meters in 0.4 seconds.) In spite of the lag, in over 7 hours of driving over surfaces like Fig. 6, the RMP never fell because of a misclassification of terrain roughness.

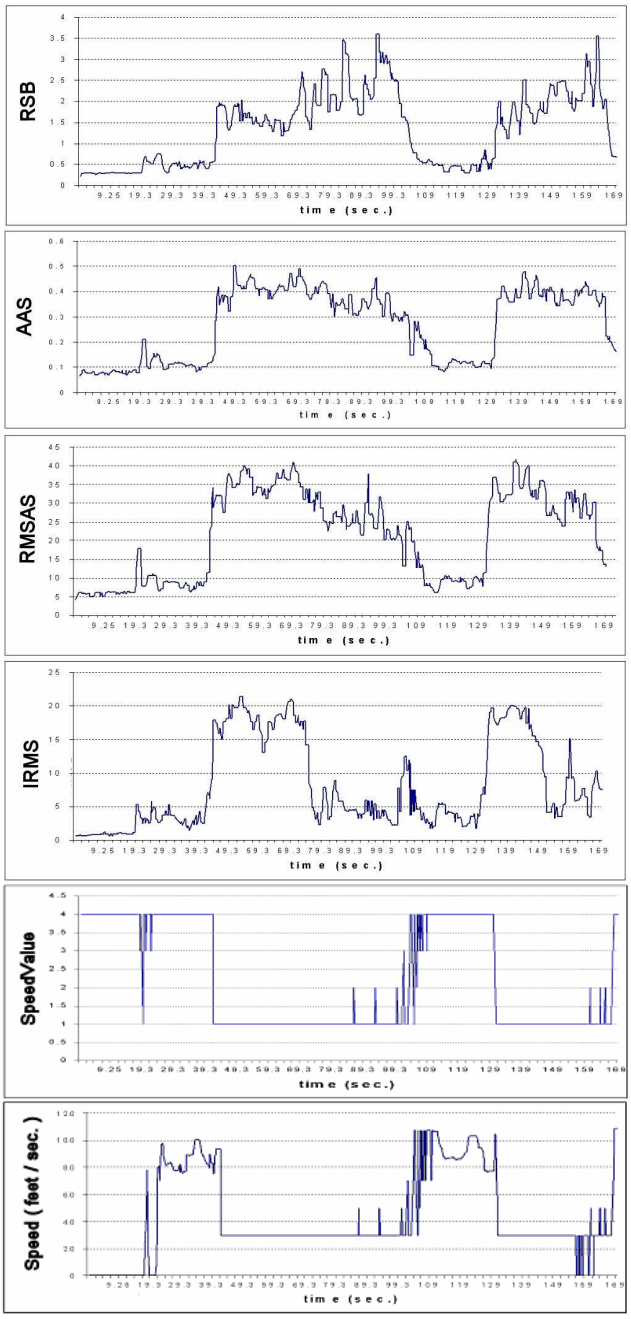


Fig. 11: Plots from the grass and asphalt experiment.

Because the RMP is always trying to balance itself, because its center of mass is very high, and because it is able to produce great velocity and acceleration, it can be considered to be one of the more difficult platforms for performing hazardous terrain detection and to take data with a top-mounted laser. For a complete rough terrain solution, future work will focus on enlarging the number of detectable terrains and adding the capability of obstacle avoidance to the system in order to allow the RMP to operate completely autonomously.

ACKNOWLEDGMENTS

This work was supported by DARPA under the Mobile Autonomous Robot Software (MARS VISION 2020) program. The Georgia Tech Mobile Robot Lab research group is gratefully acknowledged. In particular we would like to thank Yoichiro Endo and Alan Wagner for helpful and well-timed comments.

REFERENCES

[1] C. Wellington and A. Stentz, "Online Adaptive Rough-Terrain Navigation in Vegetation," Proceedings of the IEEE International Conference on Robotics and Automation, April, 2004.

[2] R. Simmons, E. Krotkov, L. Chrisman, F. Cozman, R. Goodwin, M. Hebert, L. Katragadda, S. Koenig, G. Krishnaswamy, Y. Shinoda, W.L. Whittaker, and P. Klarer, "Experience with Rover Navigation for Lunar-Like Terrains," Proceedings of the 1995 Conference on Intelligent Robots and Systems (IROS '95), IEEE, 1995, pp. 441 - 446.

[3] E. Krotkov, R. Simmons, F. Cozman, and S. Koenig, "Safeguarded Teleoperation for Lunar Rovers," 26th Int'l Conf. on Environmental Systems, July, 1996.

[4] M. Hebert, C. Caillas, E. Krotkov, I. Kweon, and T. Kanade, "Terrain Mapping for a Roving Planetary Explorer," Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '89), Vol. 2, May, 1989, pp. 997-1002.

[5] P. Bellutta, R. Manduchi, L. Matthies, K. Owens, A. Rankin, "Terrain perception for Demo III", Proceedings of the Intelligent Vehicles Symposium, Dearborn, Michigan, October 2000.

[6] S. Singh, R. Simmons, T. Smith, A. Stentz, V. Verma, A. Yahja, and K. Schwehr, "Recent Progress in Local and Global Traversability for Planetary Rovers," Proceedings of the IEEE International Conference on Robotics and Automation, 2000, IEEE, April, 2000

[7] M. Montemerlo and S. Thrun. "A multi-resolution pyramid for outdoor robot terrain perception". In Proceedings of the AAAI National Conference on Artificial Intelligence, San Jose, CA, 2004. AAAI.

[8] L. Henriksen and E. Krotkov, "Natural Terrain Hazard Detection with a Laser Rangefinder," IEEE International Conference on Robotics and Automation, Vol. 2, April, 1997, pp. 968-973.

[9] Andres Castano, Larry Matthies: "Foliage Discrimination Using a Rotating Ladar". ICRA 2003: 1-6

[10] Jose Macedo, Roberto Manduchi, Larry Matthies: "Ladar-Based Discrimination of Grass from Obstacles for Autonomous Navigation". ISER 2000: 111-120

[11] Evans, J.M., Chang, T., Hong, T.H., Bostelman, R., Bunch, W.R., "Three Dimensional Data Capture in Indoor Environments for Autonomous Navigation," NISTIR 6912, September 2002.

[12] R. Hoffman and E. Krotkov, "Terrain Roughness Measurement from Elevation Maps," SPIE Vol 1195 Mobile Robols IV, 1989.

[13] A. Talukder, R. Manduchi, R. Castano, K. Owens, L. Matthies, A. Castano, R. Hogg , "Autonomous Terrain Characterization and Modeling for Dynamic Control of Unmanned Vehicles", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Lausanne, Switzerland,2002.

[14] Georgia Tech Mobile Robot Laboratory, User Manual for MissionLab, Version 5.0, www.cc.gatech.edu/ai/robot-lab/research/MissionLab, January 2002.

[15] D.C. MacKenzie, "Design Methodology for the Configuration of Behavior-Based Robots", Ph.D. Diss., College of Computing, Georgia Inst. Of Tech., 1997.

[16] Segway Robotic Mobile Platform (RMP), Instructions for DARPA Users V1.2. Segway LLC.

[17] Jarvis, R.A. "A Go Where You are Looking Semi- Autonomous Rough Terrain Wheelchair", First International ICSC Congress on Autonomous Intelligent Systems, Deakin University, Geelong, Australia, 12-15 Feb. 2002.

[18] Homayoun Seraji, Bruce Bon: Multi-Range Traversability Indices for Terrain-Based Navigation. ICRA 2002: 2674-2681

[19] Benjamin Bau "A Report on the Segway's Performance as a Robotic Mobility Platform", www.cis.upenn.edu/marsteams/Segway/Report.pdf