Terrain Obstacle Detection and Analysis using LIDAR

Uland Wong
uyw@andrew.cmu.edu
PROSPECT Group
Robotics Institute
Carnegie Mellon University

Advisor: Dr. John Dolan

Introduction

Autonomy is crucial in planetary robotics where extreme distance and limited resource availability make continuous teleoperation impractical. Environmental obstacles represent a fundamental hurdle, as natural hazards may prevent robots from accomplishing certain tasks by invalidating path plans. In the worst-case scenario, it may even result in physical harm to the robot. Valuable resources can be saved by allowing the robot to resolve most obstacles automatically and requiring manual intervention only when absolutely necessary. Current obstacle avoidance schemes are able to detect most monolithic obstacles easily by looking for signatures like the range gradient. However, many terrain hazards are missed because the negotiability of passable terrain is rarely considered. In this paper we outline the development of an obstacle detection system for rovers used in the PROSPECT project at CMU and a general approach to terrain analysis for robots in the lunar environment. The prototype system makes use of LIDAR technology as the primary mode of sensing and specifies a one-DOF actuation for the Sick LMS 200 sensor. Data gathered from the sensor are used to analyze the terrain for obstacles and generate an internal hazard representation of the world. Predicted obstacles are then assessed and assigned a certainty and threat value which will aid the robot in navigating the environment.

Prior Work

There are many motivations for using active optical sensors for perception. Precision as well as reliability are often the cited reasons. Moreover, a significant amount of research has been conducted in LIDAR obstacle avoidance as the technology has become more affordable. However, the application of LIDAR to terrain characterization has only recently been suggested. The following research forms the foundation of the terrain detection work.

Roberts proposes a primitive fixed-mount laser range finder mounted on the front of vehicles to detect immediate obstacles and terrain hazards [3]. The sensor is able to gather terrain information only if the vehicle is moving, but the setup is very popular due to its ease of construction and use. While this is sufficient for simple obstacle avoidance, it does not contribute to autonomous robots that navigate by path planning. In order to avoid obstacles in autonomous plans, knowledge of the probable area of traverse must be obtained. The author mentions that an actuated scanner capable of generating terrain data regardless of vehicle motion is the logical progression of obstacle avoidance research.

Henriksen and Krotkov suggest that there are three primary terrain hazards detectable with laser and compare the relative detection rates between LIDAR and vision

in their paper [2]. Positive elevation hazards, also known as "steps," indicate an abrupt rise in the level of terrain. This class of hazards includes boulders, natural perforations in the landscape and broken rock surfaces. Both laser and vision perform equally well in detecting step hazards. Negative elevation hazards, called "ditches," represent abrupt downgrades in the landscape such as craters and cliffs. Laser range finding presents a drastic improvement over vision-based methods in detecting negative hazards due to difficulties in identifying distances through vision. The last class of terrain hazard is known as the "belly" hazards. These represent areas of terrain, such as dunes, that are continuous but not traversable due to the clearance physics of the robot, and are equally likely to be detected by LIDAR and vision methods.

Henriksen further suggests a general approach to detecting and classifying each hazard class using a laser scanner. The slope in the forward direction of travel is calculated and then compared against the terrain tolerances of the robot. This can be further generalized to finding the gradient in a height map generated by progressive range finding. However, the proposed algorithm is unable to detect steep continuous slopes or natural waves in the terrain which are non-hazardous but may lead to suboptimal paths of traverse. Henriksen concedes that LIDAR terrain analysis can benefit from a more sophisticated three-dimensional approach.

Finally, Vandapel suggests a method of computing saliency features for local regions in LIDAR point clouds [5]. The distribution of the saliencies is captured using the Expectation Maximization algorithm such that new data can be classified using a Bayesian classifier. Although his application is directed toward segmenting foliage and power lines in a wooded environment, there are many possible applications, including terrain analysis. Of particular importance in Vandapel's paper is the local point statistic on which he runs the learning algorithms. Principal Components Analysis (PCA) in a local neighborhood reveals certain intrinsic properties of the point cloud[1]. Specifically, the relative magnitudes of the eigenvalues (latencies) generated by PCA can be used to classify whether a point cloud is dominantly pointy, linear, or smooth. The features metrics, named pointness, curveness and surfaceness, remain stable across a variety of terrain types and are highly discriminative. The surfaceness metric in particular correlates strongly with the uniformity of point clouds forming two-dimensional manifolds.

Methodology

A general approach to the design, construction and implementation of the system is presented. Specifically, sensor design decisions are introduced and justified and a general overview of components for the system is given below.

A sense-plan-act paradigm is assumed on the prototype system, where the methodology developed will mainly occur during the sense and plan phases of navigation. Specifically, we require the robot to remain completely motionless during both phases. After the robot has ceased moving, the system will proceed to gather terrain data. The system is designed around the Sick LMS200 laser scanner, which is readily

[1] Principal Components Analysis finds a new coordinate system for a data set such that the primary axis contains the direction of greatest variation. This transformation reveals manifolds, which are low-dimensional distributions of data embedded in a high-dimensional space. This is particularly useful in obstacle detection because terrain is essentially a two-dimensional plane embedded in a three-dimensional space with some slight variation.

available and easy to use. The LMS200 gathers a plane of ranges emitted about an arc from the sensor, necessitating a powered tilt mechanism to allow volumetric scans. The system is tuned to gather 90 raster lines over 90 degrees, with each raster having a 100-degree field of view at 1-degree granularity, corresponding to a range image resolution of 90x100 pixels. A passability analysis will be performed on the resulting point cloud consisting of one or more interlaced range images to determine possible navigation routes and costs associated with each point, after which the robot can proceed again. The entire process should take an estimated seven seconds on the prototype system from stopping the robot to starting it[2]. Many mechanical and software parameters can be tuned, however, to achieve the necessary balance between usability and accuracy.

The actuated scanning mechanism was designed with the NASA K10 rover in mind. The assembly is mounted at an appropriate height (1 meter) on the front of the robot to allow sufficient horizontal range in the unsaturated portions of the scanning arc. Mechanical constraints limit the tilting angle to -75 degrees and +15 degrees on the robot axis; however, most data gathered above the horizon are meant primarily for near-obstacle avoidance.

Low-level positioning is accomplished by an embedded computer running a closed-loop control and accepting data acquisition commands from a master computer. Communication to the LMS200 sensor itself is managed directly from the master computer. A laptop computer that runs high level algorithms on the robot is responsible for initiating the scan process, synchronizing incoming packets with sensor pose and processing the laser range data. It will then use that information to plan a path for the robot.

The terrain analysis software accepts a laser range image consisting of points in Cartesian space. The analysis attempts to discretize the entire scan range into an occupancy grid. Each square is analyzed by the algorithm and receives a certainty value and also an assessed threat value. Using these two values, a cost function for the navigation software can be derived.

Mechanical System Design

Design constraints for the tilting apparatus included structural rigidity for precise operation on a mobile robot, reliability for continued use throughout a long mission lifetime, and small payload weight and power footprint. Additional design factors included data quality and uniformity, and easy integration on the K10 platform. These in turn were dependent on motor size and quality, available payload capacity, and available power/voltage limits on the robot. All of these factors were considered in the mechanical design of the system.

Analysis of currently implemented designs and prior experience led to a decision for belt-driven actuation through a highly geared dc motor. Dual encoders mounted on the axis of rotation and the motor output shaft provide feedback control for global position and velocity. This setup strikes a good balance between motor/mechanism weight, backlash, power consumption and ease of construction. The high gear ratio eliminates the need for a larger motor, which would dramatically increase the weight and power consumption of the system. Additionally, it simplifies controls to the point that a proportional or PD loop suffices instead of a full PID control law. Using a belt drive as

[2] This number is a rough estimate factoring calculated scan time and reasonable computation time.

opposed to direct gearing decreases weight, allows freer placement of the motor and also reduces control complexity. Though a timing belt requires maintenance and is less reliable than gears, it is believed to be adequate for the specified operational requirements.

Selection of the motor was influenced by the desired resolution and available data rates from the LMS200. Specifically, the throughput of the sensor at one-degree granularity is 13.3ms/scan. Completing 90 scans to produce the range image would therefore require a minimum of 1.2 seconds. However, future experiments may make full use of the LMS200's precision, which would require 4.8 seconds to complete a voluminous scan[3]. The motor must therefore be able to actuate the proper load (the sensor weighs 4.5Kg) from 18 deg/s to 75 deg/s. It was determined that a Maxon RE25 motor with 531:1 gearing was sufficient for the assembly.

Motor	**Gearing**	**Voltage(Max)**	**Power Rating**	**Encoder**
Maxon RE25	531:1	48V	20W	HEDS5540

Design of the tilt platform was based on a prior design by Ben Brown and Arne Suppe at CMU for use on Segway robots. The design was adapted for horizontal mounting on the K10 robot and to accommodate our specific motor dimensions.

The design specifies two 1/8" mounting plates to be attached to the LMS200, creating an axis of rotation as close to the center of mass as possible without obstructing the sensor window. The laser pivot is 4.0" below the top of the assembly and extends 8.3" forward from the mounting surface to the robot. The assembly attaches to two 80/20 frame members on the front face of the robot through four 1.5"-high cylindrical spacers. A sprocket assembly on the left side of the axis of rotation provides tilt actuation through a belt drive connected to the driving sprocket located on the motor. The motor itself is placed to the rear of the device to reduce obstruction. The motor is attached with an adjustable mounting plate to tighten the belt if necessary. The axial encoder is mounted on the right side of the device opposite the sprocket. Finally, a crossbar provides structural support and also serves as a mechanical stop for the sensor.

[3] The LMS200 has a maximum resolution of 0.25 degrees in a 180 degree arc, providing a possible 720 points per line. The current motor setup can realistically position to within 0.01 degrees, providing a maximum theoretical resolution of 9000 x 720 per scan. Practical limits the on scan duration, however, restrict the actual resolution to much less.

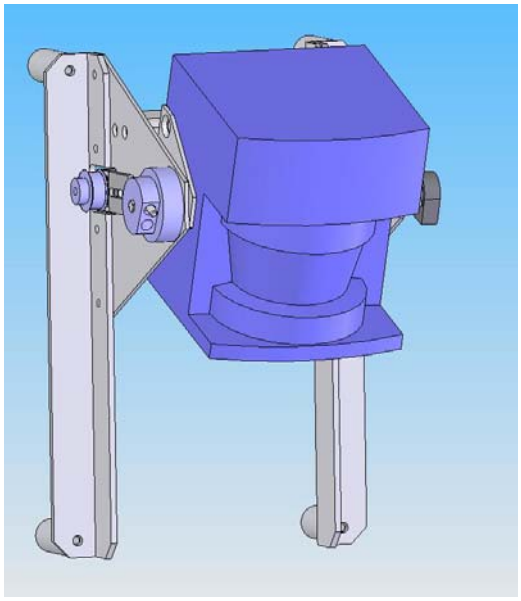


Figure 1. Tilt Mechanism Frontal View

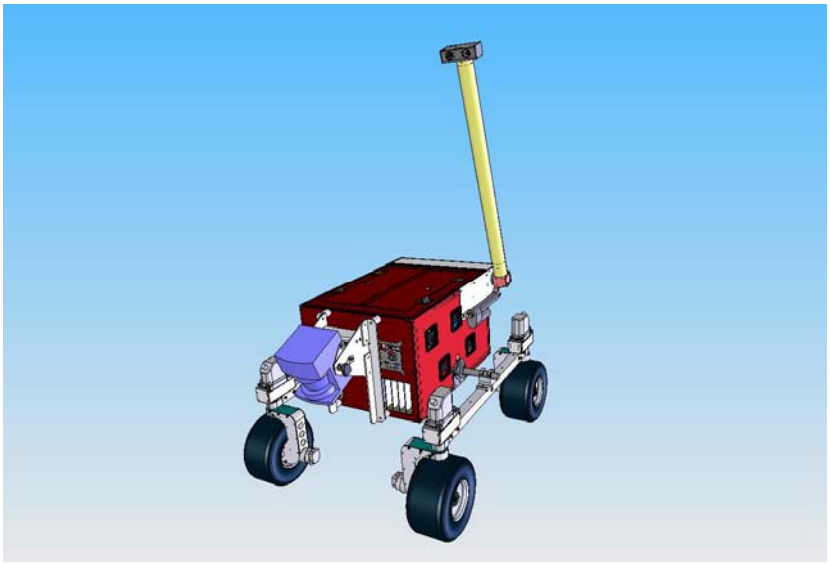


Figure 2. Concept Design on K10 Robot

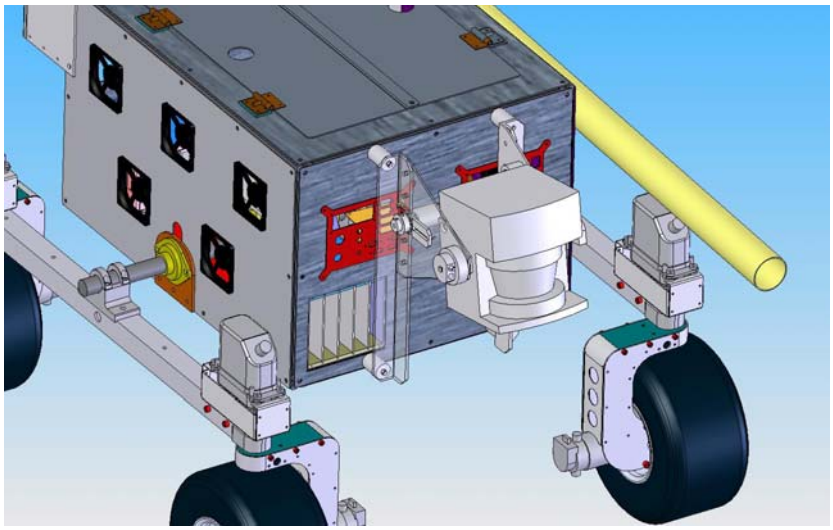


Figure 3. Close-up of Mounting Configuration

Hardware Interface

The system contains three primary functional units: the master computer, the LMS200, and the tilting mechanism and support electronics. Great care was taken to ensure that all components would be maximally compatible in terms of power requirements and communications.

RS-232 was chosen as the primary mode of communication between the major components. The laptop's lack of dedicated digital I/O, the prevalence of USB-to-Serial converters and the ease of programming RS-232 serial ports offset any possible speed and reliability issues. Reliability problems were especially unlikely since there was not a significant distance between the laptop and the other components. Additional factors for choosing the protocol were the lack of optional RS-485 hardware for the LMS200 and difficulty in implementing RS-485 on a microcontroller. Please refer to Figure 4 for a system communications diagram.

The only available voltage level on the K10 was 12V DC, which could be regulated to 5V for most TTL applications. However, another voltage level was also required. 24V seemed to be the most sensible compromise between the LIDAR, which requires 24V ± 5%, and the DC motor, which requires 12-48V. Thus, a DC converter was used to transform the 12V output of the robot to the necessary 24V level. Please refer to Figure 5 for a complete electrical schematic.

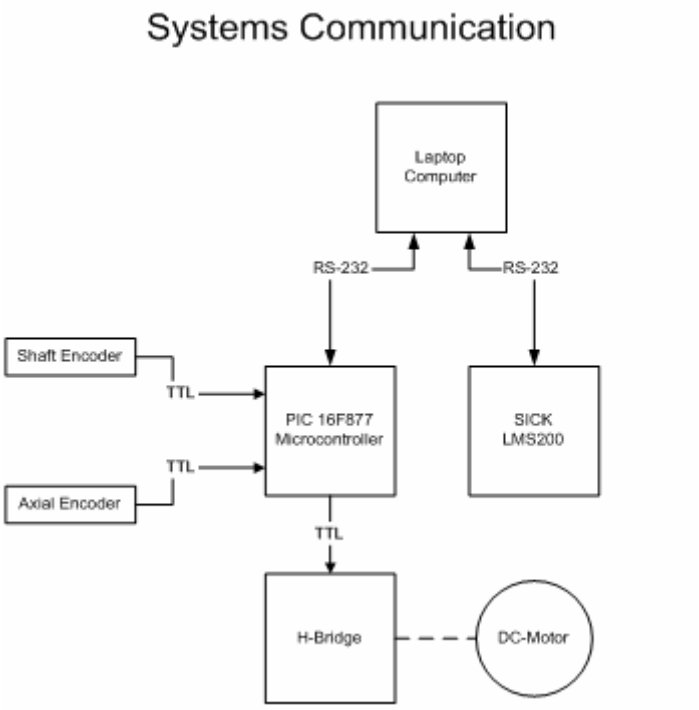


Figure 4. Communications Schematics

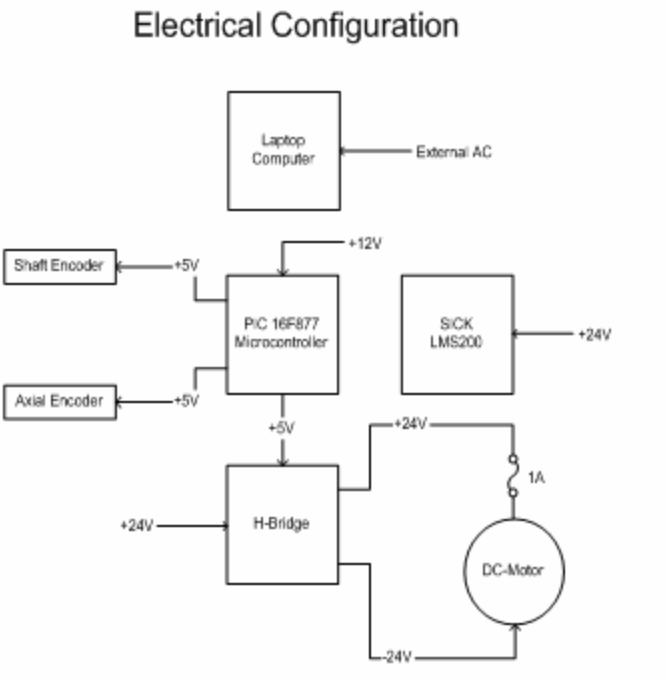


Figure 5. Electrical Schematics

The dedicated motor controller was written in C and implemented on a PIC16F877 microchip using the PICC compiler. The microcontroller is responsible for powering and driving the motor through a commercially available H-bridge circuit and receiving feedback from two encoders. Additionally, it must receive commands from the master computer and also return position information over serial. The PIC16F877 was selected because it is inexpensive and readily available, and because the author has prior experience.

The limited size and speed of the microchip restricted the number of features that could be implemented. A more accurate acceleration profile, software current limitation and full PID control were among the features unavailable due to code size constraints. Fortunately, none of them are particularly important to the functionality of the system.

Microcontroller	**EEPROM**	**RAM**	**Frequency**
PIC16F877	8KB	384B	4MHz

A proportional homing control was written in the microchip's main program loop with interrupt-driven input from the master computer over serial. Once a position is received, the interrupts are disabled and the program relinquishes control to the main PD loop. During the motion control phase, the tilt position is queried regularly and used to adjust the current in the motor as well as send position updates over serial to the computer. After the motor has arrived at the desired position to within a certain threshold, an acknowledgment is sent to the laptop, and the idle homing loop once against assumes control.

The encoders are connected to the PIC's two 16-bit built-in timing circuits using the external edge trigger. Quadrature encoding is not currently supported; instead, a total position count from the motor shaft encoder is heuristically corrected with the global position output from the axial encoder. Control of the motor current is accomplished by connecting the output enable pin on the H-Bridge to a CCP output from the microchip. By varying the pulse-width output at an appropriate granularity, motor velocity can be precisely controlled. Motor direction is reversed by toggling pins on the H-Bridge circuit to reverse the polarity of the motor's power lines. Please refer to Figure 6 below for a detailed diagram of the motor controller dataflow.

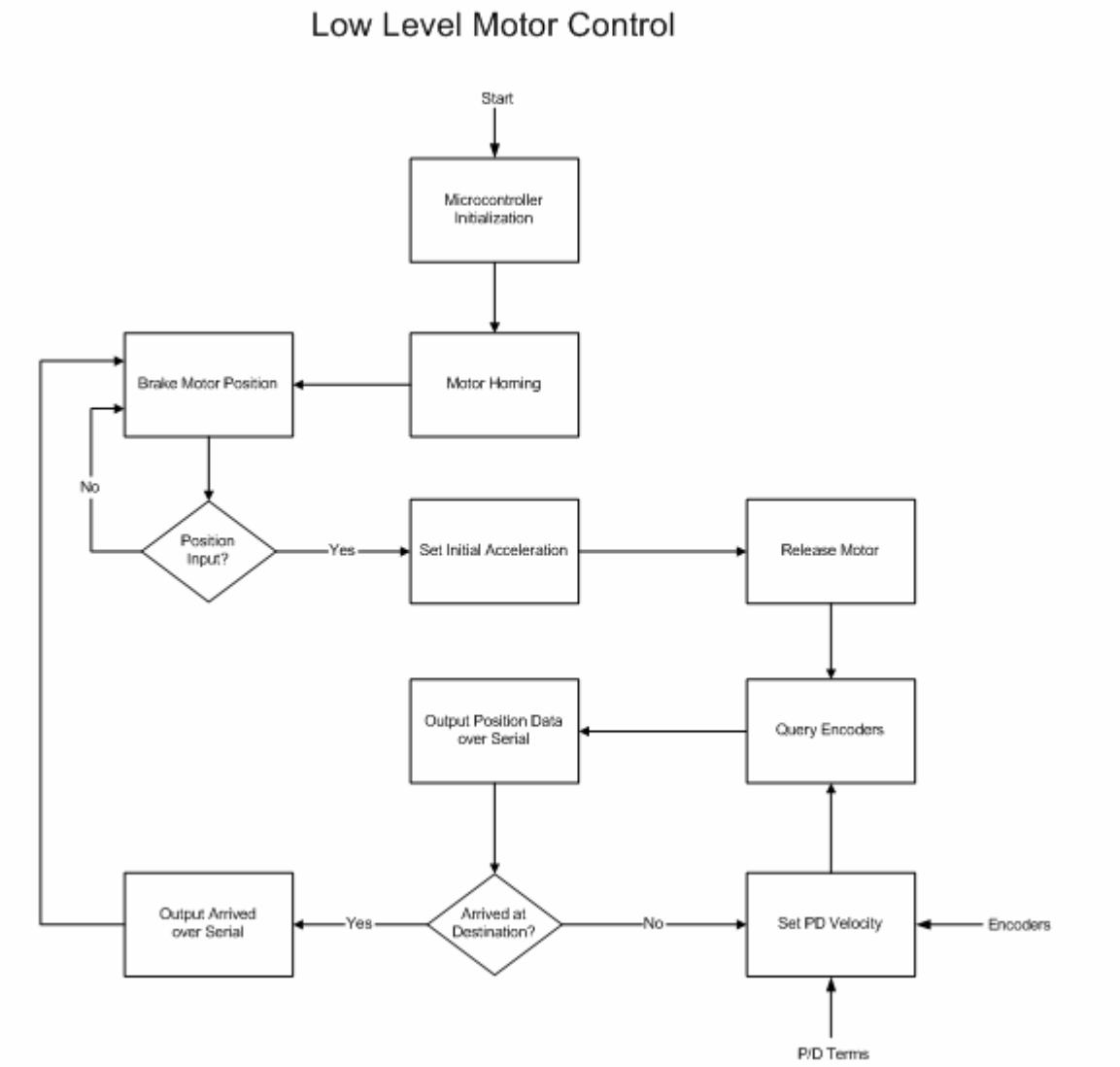


Figure 6. Low Level Controls

Terrain Analysis Software

Traditional gradient and continuity-based methods are able to reliably detect large obstacles with limited computational power. However, they lack the discriminability to fully classify a wide variety of terrain types. Conversely, dimensionality reduction methods mentioned by Vandapel [5] are able to provide a qualitative analysis of terrain, but are incapable of making dependable binary decisions about obstacles. It is hoped that by combining gradients with PCA to analyze the roughness of terrain, a robust system can be developed that not only detects obstacles but can also quantify the overall navigability of different landscapes.

To ensure that the PCA method works as well with mostly planar terrain data as point clouds from vegetation which are less likely to be true two-dimensional manifolds, the latent values of 10,000 "patches" of simulated terrain were plotted against the known surface variances of each patch. As expected, the correlation between surface roughness and the magnitude of the least significant eigenvalue is high, and the relationship is

mostly linear (Figure 7). However, the experiment does not show the true robustness of the dimensionality metric. While the variance of terrain data is a quick and direct measure of surface roughness, it is only accurate and suitable for terrain that is intrinsically parallel to the robot. Thus it was only possible to test terrains with non-varying mean height. In addition to performing well in the planar case, the dimensionality metric will correctly quantify any manifold in three-space, including slopes, cliffs and mounds. We will use the smallest latent value from PCA as the primary terrain descriptor in the algorithm, which we will call the *roughness metric*.

In order to run PCA on real terrain data, the point cloud must be discretized into windows representing a patch of terrain. This also facilitates a significant computational speedup to using a k-nearest neighbor approach in both dimensionality reduction and running the gradient algorithm. The process, known as "binning," divides the total terrain data into a 100x100 grid of equally sized rectangles, using the mean height value of member points to represent the entire bin when necessary[4]. A roughness descriptor and a certainty value are generated by running PCA on each bin using the member points.

A gradient descriptor is also calculated for each bin. The gradient detects the relative height difference between one patch of terrain and all of its neighbors. This provides accurate detection of large obstacles and terrain continuities but is often too conservative to find all true obstacles. Nevertheless, this descriptor is useful as areas marked as hazards by the gradient method can be immediately discounted from any further computation. Our specific implementation of binary obstacle detection assigns the value of the largest gradient from a patch to one of its neighbors as the cost for that patch. We call the gradient descriptor the *continuity metric*. Figure 8 shows typical terrain data gathered by the sensor. Figure 9 shows the relative costs corresponding to each location in the terrain. Elevation in the z-axis corresponds to a greater cost for navigating that location.

An initial "high danger" decision is made based on an empirically chosen threshold of 10cm on the gradient descriptor, where there are an adequate number of bins such that their centroids are 10cm apart. Any locations containing a grade or downgrade steeper than 10cm are marked as impassable. The roughness values for the rest of the bins are then normalized, and a roughness threshold of 0.5 is applied to find any remaining hazards[5]. The passable points are the intersection of the points left unmarked by the roughness metric and the continuity metric. The cost for traversing each location is given by the relative value of the roughness indicator and the certainty of analysis at that point is given by the number of points in the bin.

[4] The number of bins is arbitrary, so long as there are enough points for the numerical stability of PCA. Empirically, an average of about 20 points per bin is desirable. For a coarse point cloud, such as that resulting from a 1-degree scan, 20x20 bins is a good number. Most of the analysis in this paper was completed using two interlaced range images of 180 lines at 0.5 degree resolution.

[5] The 0.5 roughness threshold means that we wish only to consider the easiest 50% of terrain as passable. This is a primitive metric that works to make a binary decision about obstacles; however, most applications will probably benefit from a more intelligent approach to terrain analysis using the unfiltered cost map directly.

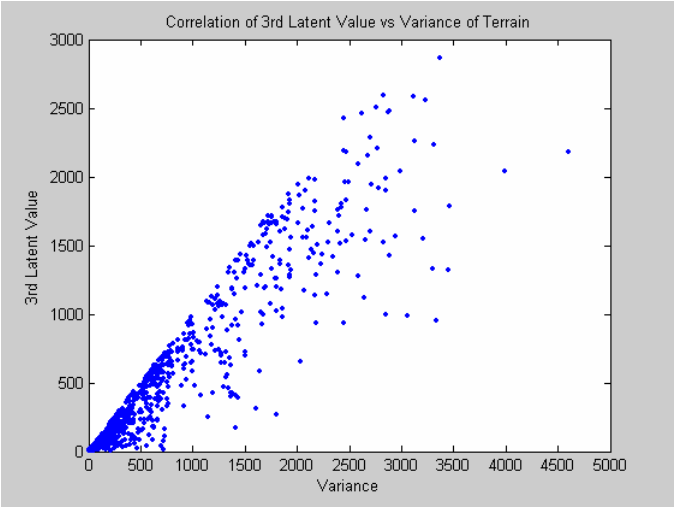


Figure 7. Correlation of Latent Values

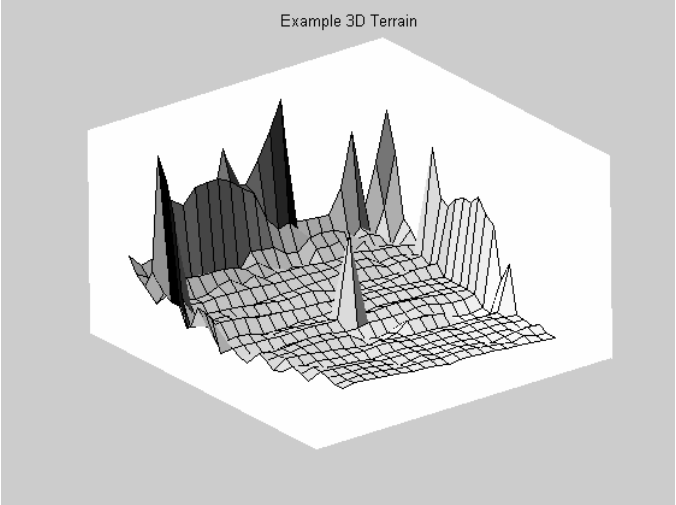


Figure 8. A Typical 3D Terrain from a Point Cloud

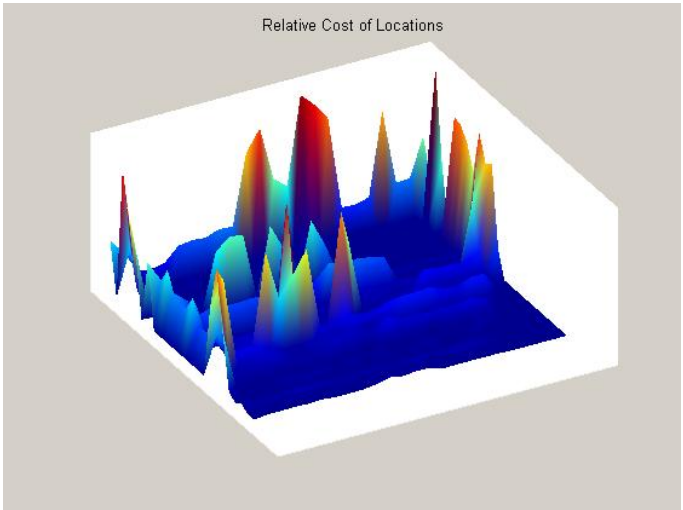


Figure 9. Relative Costs of each Location in the Terrain

The high-level terrain analysis software is realized in Matlab. The primary reason for choosing an interpreted language is because Matlab supports rapid development and debugging of software and allows the programmer to concentrate on the algorithmic aspects of the research.

The terrain analysis software comprises two main modules and a library of supporting functions. The first module packages all obstacle detection and analysis software and acts as the master routine for the system. The data acquisition package is responsible for coordinating interaction between the LMS200 and the tilt assembly to generate the range image. Lastly, the support library includes functions for principal components analysis, serial I/O and data visualization.

The process starts when the robot requests a cost map for the current terrain. This prompts a startup procedure in which the high-level routine requests a point cloud to be generated. Control is transferred and the data acquisition system begins by homing the tilt unit. After an acknowledgment that the sensor has finished moving, the system requests the motor to move to the initial scan angle. Upon receiving the ACK signal, planar range data are read from the LMS200 and saved. The exact value of the encoders is also sent to the master computer from the microcontroller without request. This is to provide as accurate a position as possible within the threshold for the motor settling. The motor is then incremented by the predetermined step amount (which is dependent on the resolution of the range image) and the process is repeated until the desired number of vertical scans has been received. The resulting point cloud is sent back to the high-level routines.

The data at this stage, however, are misaligned with respect to the world frame. The LMS200 outputs range measurements and a corresponding yaw angle. When the sensor is tilted, another angular degree of freedom, pitch, is introduced. However, a standard spherical coordinate transform will produce an incorrect output, as sensor pitch and measurement angle are actually dependent axes in the world frame. To solve this, each set of planar scans is mapped to the correct Cartesian (x,y,z) triple from $(\rho,\theta,0)$ using a three-dimensional rotation matrix through the angle φ. The matrix, pictured in Figure 11, rotates any point φ-degrees counterclockwise about the x-axis in the world frame.

With the corrected point cloud, the system begins the analysis phase. First, the points are binned into the occupancy grid. The first pass of the algorithm calculates the gradient of the cloud at the grid-level and any resulting "high danger" hazards. The second pass iterates through each bin individually and finds the roughness features from principal components analysis. Finally, using the gradient value and the salient features, the cost of the grid is determined and a certainty is assigned. Once all the bins have been analyzed, the cost map is completed.

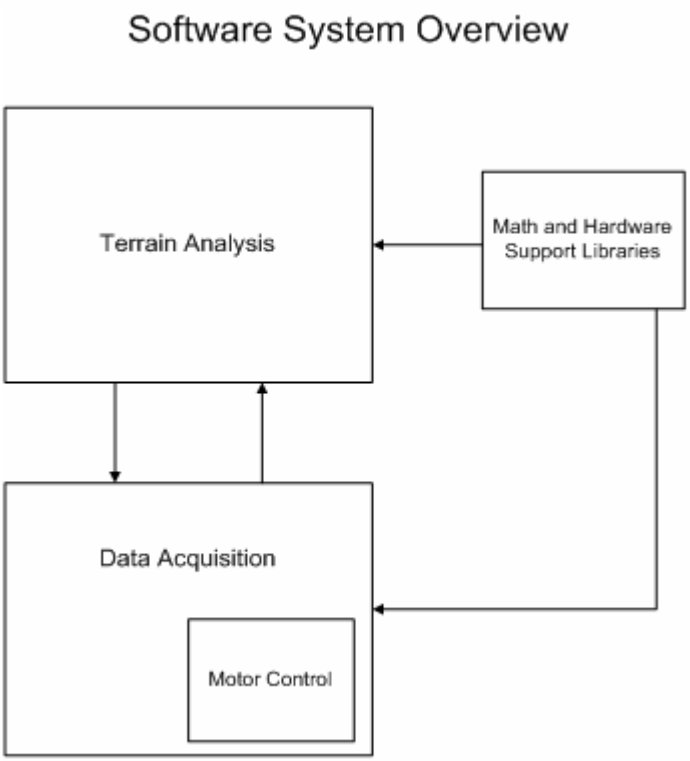


Figure 10. Software System Block Diagram

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{pmatrix}$$

Figure 11. Rotation Matrix

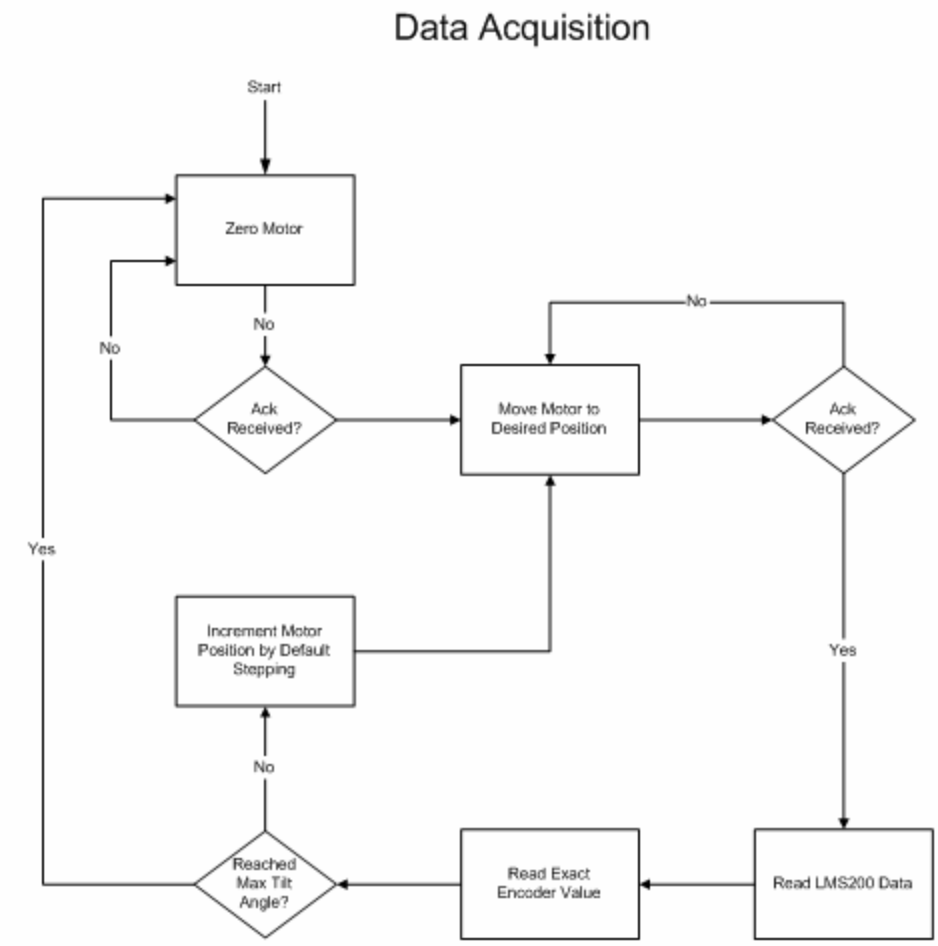


Figure 12. Data Acquisition Control Flow

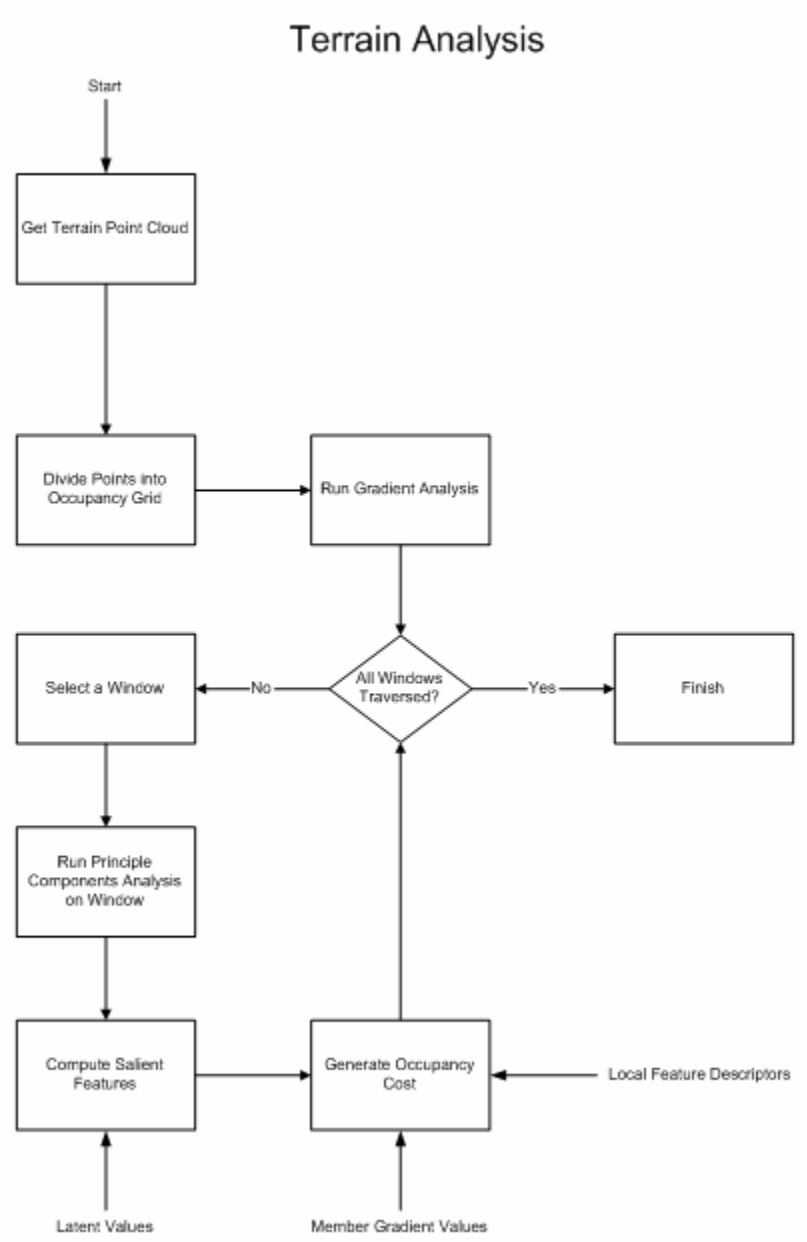


Figure 13. Terrain Analysis Control Flow

Results

The system was tested primarily in an office environment, where obstacles consisted of chairs, people and other common robot-sized objects. Rough terrain was simulated by the selective placement of rumpled cloth. A variety of data was gathered at each scene to determine the aforementioned empirical system constants. The following figures show a thorough analysis of one such scene using the phases in the proposed algorithm. The scene shows, from roughly a 0.5-meter perspective, rough terrain placed in front of a door (shown to the very north in Figure 14) leading out of a small room with a person standing in front of the robot. Portions of the terrain where there are no readings due to low-reflectivity materials or occlusions are interpolated bilinearly from surrounding points to create a continuous data set for analysis.

Figure 14 is simply a convenient reconstruction of the point cloud into a surface map using Delaunay triangulation. The data shown have already been discretized into 100x100 bins. Figure 15 shows the magnitude of the gradient of the example terrain. Lighter areas represent a greater difference in height between a location and its neighbors and dark areas represent no difference. Figure 16 shows the results of the continuity metric to detect "high danger" regions, representing large hazards. Lastly, Figure 17 shows the results of the roughness metric using PCA.

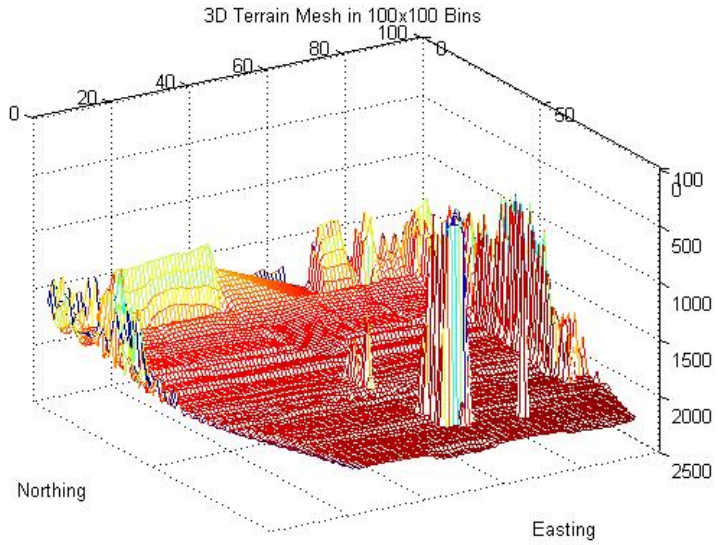


Figure 14. Sample Discretized 3D Terrain

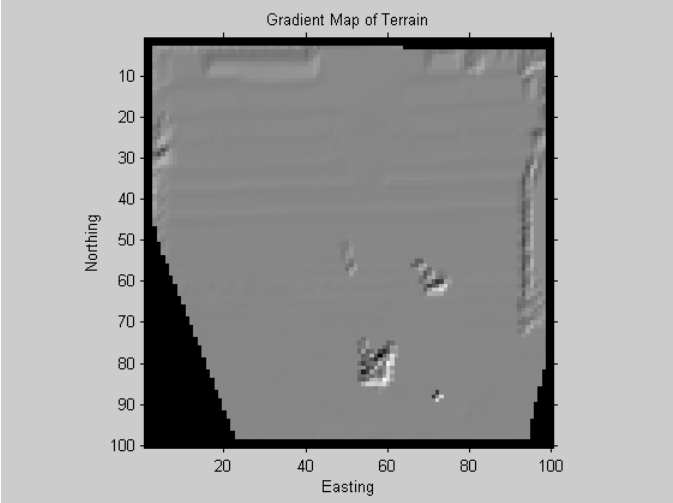


Figure 15. Gradient Map of Terrain

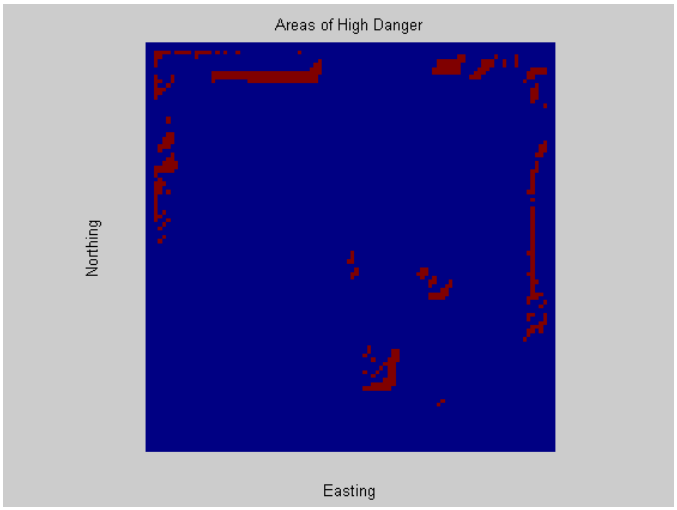


Figure 16. Obstacle Detection by Thresholding Gradients

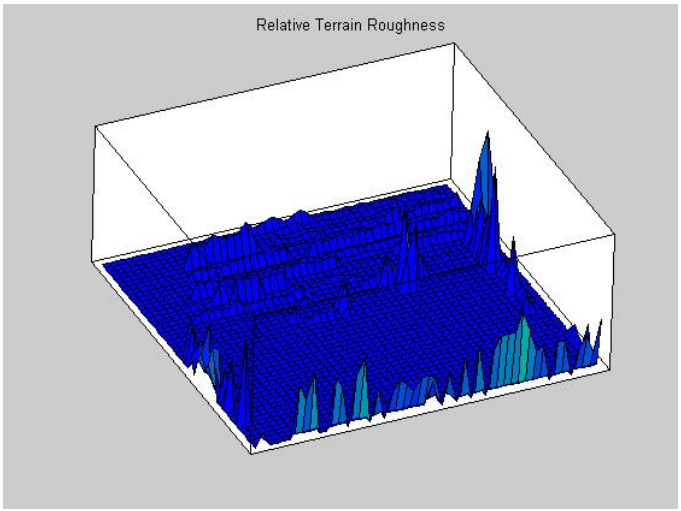


Figure 17. Terrain Analysis
Note: The spike artifacts near the edges are from PCA instability due to a lack of data.

Recommendations for Future Work

There are several areas where improvement is possible. Most are contingent on advancing the system from prototype to final model. First, moving the code from Matlab emulation to a compiled executable, perhaps even changing the language of implementation, is of primary importance[6]. While Matlab allows rapid prototyping of complex software systems, it is fundamentally unfit to run real-time field applications. Second, the motor control system leaves some things to be desired. It is sufficient for the application, but is ultimately a temporary solution. An embedded computer such as a PC104 should be purchased and dedicated entirely to motor control. This will help alleviate the load on the master computer which previously handled part of the loop. Additionally, the dedicated computer will allow maximum response and precision in positioning the instrument. Finally, a more efficient method for calculating the

[6] Matlab allows compilation of script files. However, the speed, stability and maintainability of code generated by the mcc compiler is questionable.

descriptors and assigning values is desired to speed up the algorithm. Though in testing, the latency of running PCA on each bin is small compared to the latency of reading from the serial port in Matlab, once the application is compiled, the author believes this will be the computational bottleneck.

Aside from making improvements to the system, the primary thrust of future research will be to integrate the algorithm and resulting cost map with actual planning software on the robot and to perform testing in more representative environments. By observing how planners interact given the cost information, the system can be tuned for optimal performance in the field where unexpected conditions are bound to arise. The secondary goal of future work is to continue to move toward terrain classification by matching physical characteristics of point clouds against a database of terrain types. Hopefully, using learning or other classification techniques, robots will automatically be able to determine areas suitable for construction, prospecting and even mining in the lunar environment.

Acknowledgements

Special Thanks to John Dolan who advised the project, Ben Brown for designing and machining the tilt assembly and Luis Navarro and the rest of the PROSPECT group at Carnegie Mellon University for their support and guidance.

References

[1] C. Grindle, M. Lewis, R. Glinton, J. Giampapa, S. Owens, K. Sycara. Automating Terrain Analysis: Algorithms for Intelligence Preparation of the Battlefield. In Proceedings of Human Factors and Ergonomics Society 48th Annual Meeting. 2004.

[2] L. Henriksen and E. Krotkov. Natural Terrain Hazard Detection with a Laser Rangefinder. In Proceedings of IEEE International Conference on Robotics and Automation. 1997.

[3] J. Roberts and P. Corke. Obstacle Detection for a Mining Vehicle using a 2-D laser. In Proceedings of Australian Conference on Robotics and Automation. 2000.

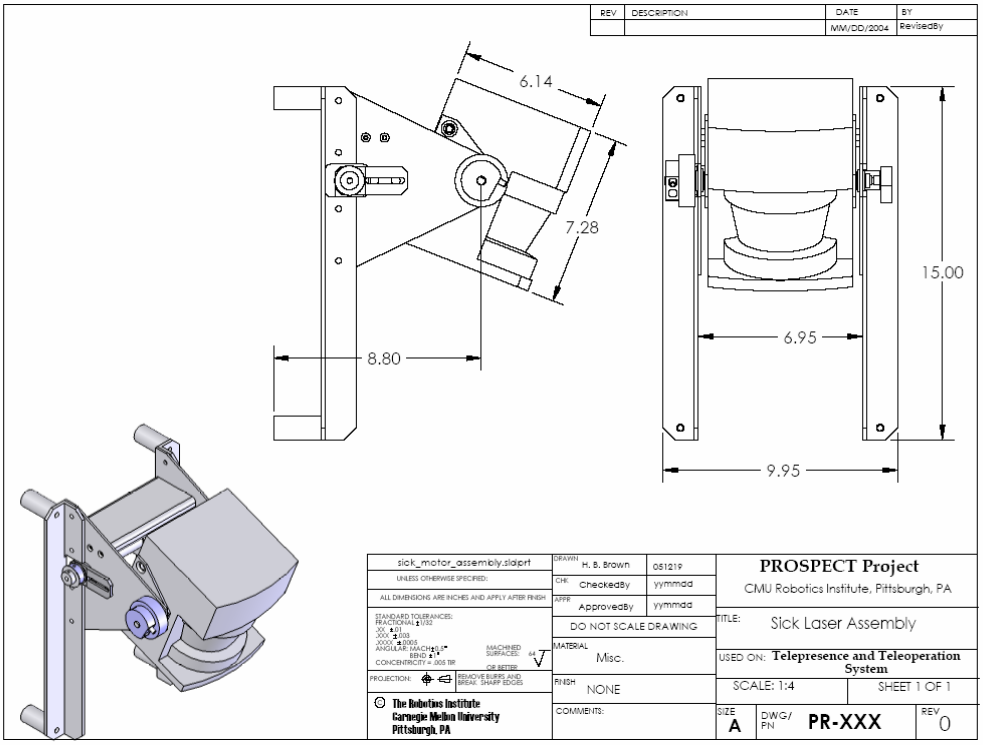
[4] H. Schuster. Segmentation of Lidar Data using the Tensor Voting Framework. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences. 2004.

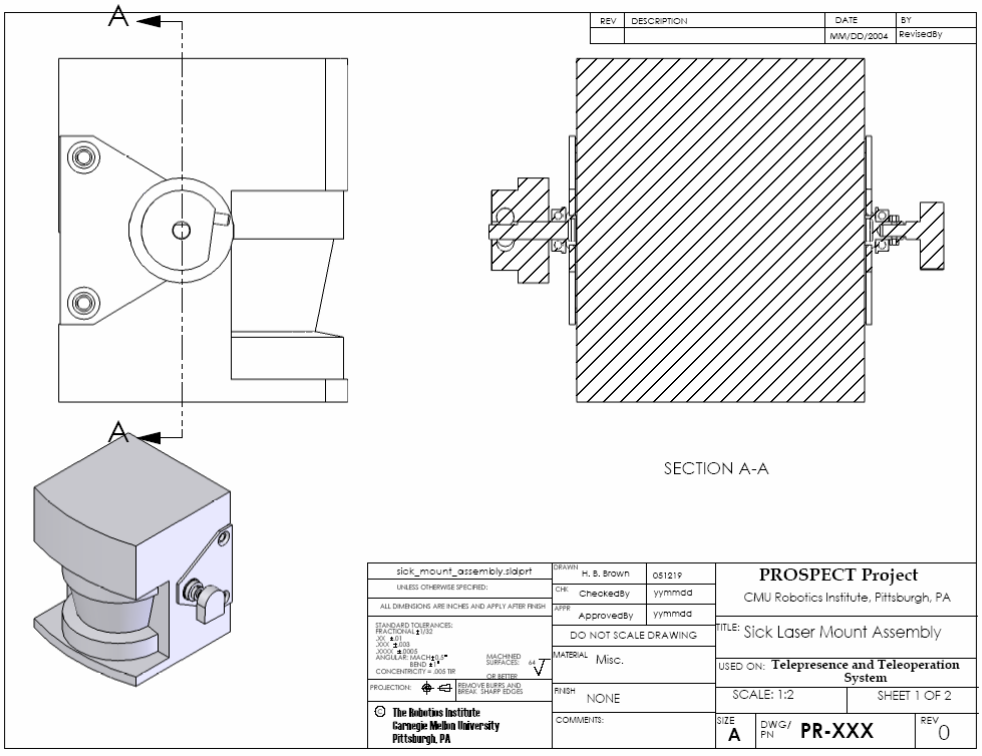
[5] N. Vandapel, D. Huber, A. Kapuria, M. Hebert. Natural Terrain Classification using 3-D Ladar Data. In Proceedings of IEEE International Conference on Robotics and Automation. 2004.

Appendix A. Parts List

Part	Notes	Cost
Motor	RE25, Maxon# 118755	$245.65
Planetary Gear	Metal 531:1, Maxon# 166180	$300.25
Shaft Encoder	HEDS5540, 500count, Maxon# 110511	$105.85
Microcontroller	Microchip PIC16F877	$2.00
Dev Board	Provided by CMU Mechatronics	$35.00
H-bridge Circuit	L298N	$4.19
Laser Scanner	Sick LMS200	$3275.00
	Total	**$3967.94**

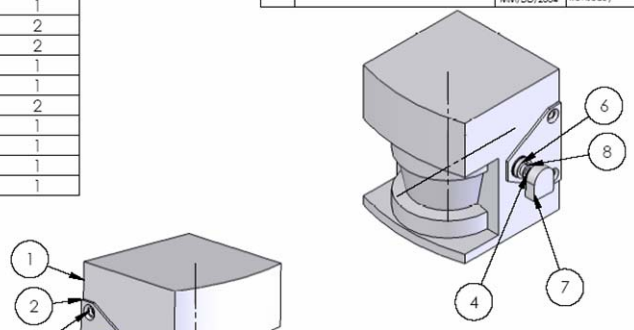
Appendix B. Mechanism Dimension Drawings

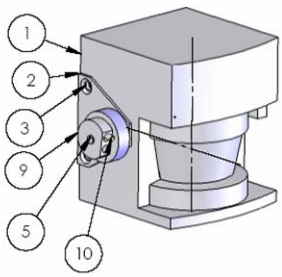




ITEM NO.	PART NUMBER	QTY.
1	sick_lms291	1
2	sick_mtg_plate2	2
3	sick_mtg_plate_spacer	2
4	encoder_side	1
5	drive_side	1
6	brg_375x875x281_fl	2
7	encoder_USD_S1	1
8	pot_screw_ring	1
9	driven_sprocket1	1
10	sprocket_cotter	1

REV	DESCRIPTION	DATE	BY
		MM/DD/2004	RevisedBy





sick_mount_assembly.sldprt	DRAWN H. B. Brown	051219	**PROSPECT Project** CMU Robotics Institute, Pittsburgh, PA
UNLESS OTHERWISE SPECIFIED:	CHK CheckedBy	yymmdd	
ALL DIMENSIONS ARE INCHES AND APPLY AFTER FINISH	APPD ApprovedBy	yymmdd	TITLE: Sick Laser Mount Assembly
STANDARD TOLERANCES: FRACTIONAL ±1/32 .XX ±.01 .XXX ±.005 .XXXX ±.0005 ANGULAR: MACH ±1° BEND ±1° CONCENTRICITY = .005 TIR	DO NOT SCALE DRAWING		USED ON: **Telepresence and Teleoperation System**
MACHINED SURFACES: 64 7 OR BETTER	MATERIAL Misc.		
PROJECTION: REMOVE BURRS AND BREAK SHARP EDGES	FINISH NONE		SCALE: 1:2 SHEET 2 OF 2
The Robotics Institute Carnegie Mellon University Pittsburgh, PA	COMMENTS:		SIZE A DWG/PN **PR-XXX** REV 0