2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Ground Plane Identification Using LIDAR in Forested Environments

Matthew W. McDaniel, Takayuki Nishihata, Christopher A. Brooks,
and Karl Iagnemma

***Abstract*—To operate autonomously in forested environments, unmanned ground vehicles (UGVs) must be able to identify the load-bearing surface of the terrain (i.e. the ground). This paper presents a novel two-stage approach for identifying ground points from 3-D point clouds sensed using LIDAR. The first stage, a local height-based filter, discards most of the non-ground points. The second stage, based on a support vector machine (SVM) classifier, operates on a set of geometrically defined features to identify which of the remaining points belong to the ground. Experimental results from two forested environments demonstrate the effectiveness of this approach.**

I. Introduction and Related Work

Unmanned ground vehicles have demonstrated effective autonomous operation in a variety of scenarios, including cross-country and urban environments [1],[2]. Future applications for UGVs will require autonomous operation in forested environments. For UGVs to plan safe paths in forested environments, the systems will need to be able to identify contours of the load-bearing surface (i.e. the ground). This will enable the system to identify positive and negative obstacles, and assess the traversability of candidate paths. This paper addresses the problem of identification of the ground in 3-D point clouds sensed using LIDAR.

Identification of the load-bearing surface from point clouds is straightforward in many 2 1/2-D environments (e.g. the surface of Mars) where most of the sensed points lie on the load-bearing surface, or in urban environments where the road surface is nearly flat. This problem becomes more challenging in forested environments, where tree trunks and shrubs occlude distant terrain and the surface of the ground is uneven.

Previous work on 3-D ground plane estimation can be grouped into techniques designed for airborne LIDAR (e.g. [3],[4]) and techniques developed for LIDAR mounted on ground vehicles (e.g. [5]-[10]). Due to the difference in perspective, algorithms targeted towards airborne LIDAR may not be appropriate for implementation on ground-vehicle-mounted LIDAR. Some of the concepts, however, may have utility in both problem domains. For example, in [3], a method is described for defining a digital elevation model. The method first creates a surface below all of the LIDAR points, then deforms that surface upwards so that it coincides with points suspected to be ground. Fig. 1 shows an illustration of such a deformed surface.

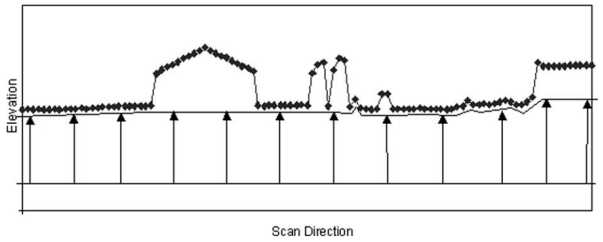

Fig. 1. Connecting lower surface to possible ground points (from [7]).

Other work has addressed ground plane estimation from UGV-mounted LIDAR data. In [5], ground plane estimation was accomplished through application of the RANSAC algorithm [11]. An approach to ground plane estimation based on Markov random fields ([8] and [9]) has been applied in environments with flat ground partially covered by vegetation. It is unclear whether such techniques would be effective in 3D forested environments.

The approach to ground plane estimation presented in [10] uses an octree framework to find best fit planes in each region in which the LIDAR point density exceeds a pre-defined threshold. A merging process is then used to connect neighboring planes which have similar normal vectors.

A work that is closely related to the method described in this paper is the ground filtering approach presented as part of a 3-D object classifier in [7]. This approach operates on the assumption that the ground will be less steep than a pre-defined slope. Following this assumption, for a candidate LIDAR data point to belong to the ground, a downward-oriented cone with vertex at the candidate point (with an included angle proportional to the pre-defined maximum ground slope) must be empty of other points. The approach proposed in this paper adopts a similar approach as part of a larger ground plane estimation framework.

This paper presents a two-stage approach for ground plane identification. The first stage is a local height-based filter, inspired by [3], which eliminated from further consideration most points that lie above the ground plane. The second

This work was supported by the U.S. Army under grant number W911NF-09-1-0334. M. W. McDaniel, T. Nishihata, C. A. Brooks, and K. Iagnemma are with the Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: mcdaniel@mit.edu, nishihat@mit.edu, cabrooks@alum.mit.edu, kdi@mit.edu). Correspondence should be directed to M. W. McDaniel (phone: 617-715-2648).

stage employs eight geometrically derived features in a SVM classifier to classify the remaining points as ground or non-ground.

This paper is divided into five sections. Section II presents the proposed approach for ground identification based on the local height-based filter. Section III presents details of the experimental studies used to validate the proposed approach. Section IV presents numerical results, and Section V presents conclusions.

II. Proposed Approach

A. Overview of proposed approach

The approach proposed here divides the task of ground plane identification into two stages. The first stage is a local height-based filter, which encodes the fact that, in any vertical column, only the lowest point may belong to the ground. In practice this eliminates a large percentage of non-ground points from further consideration. (In the test data sets presented here, approximately 98.7% of the data points were eliminated through this method.) The second stage is a SVM classifier, which combines eight heuristically inspired features to determine which of the remaining points belong to the ground plane.

B. Detailed description of proposed approach

Given a set of range data points in Cartesian space, the goal of ground plane detection is to identify which of those points belong to the ground. In this work, candidate points are represented in an inertial frame with coordinates (x,y,z).

In the first stage, the points are divided into 0.5-m by 0.5-m columns based on their x and y values. These columns are identified by indices (i,j), where $i=\lfloor x/0.5 \rfloor$ and $j=\lfloor y/0.5 \rfloor$. Thus, a point located at (2.9m, 4.1m, 1.7m) will be located in column (5,8). In each of these columns, only the lowest point (i.e. the point with minimum z) is retained as a possible ground point. For simplicity, the lowest point in column (i,j) is hereafter denoted $P_{i,j}$, and its coordinates are referred to as $(x_{i,j}, y_{i,j}, z_{i,j})$.

In the second stage, a variety of features are used to represent attributes of each point $P_{i,j}$ and the lowest points in each of the eight neighboring columns (i.e. $P_{i-1,j-1}$, $P_{i-1,j}$, $P_{i-1,j+1}$, $P_{i,j-1}$, $P_{i,j+1}$, $P_{i+1,j-1}$, $P_{i+1,j}$, and $P_{i+1,j+1}$), as shown in Fig. 2. A set of eight features was selected from a larger set of possible features based on their usefulness in discriminating ground from non-ground. These features, denoted $f_1,\ldots,f_8$ are combined into a feature vector $F_{i,j} = (f_1,\ldots,f_8)$ for each point, which is used by a classifier to identify whether that point belongs to the ground. These features include:

- f_1: Number of occupied columns in the neighborhood of column (i,j)
- f_2: Minimum z of all neighbors minus $z_{i,j}$
- f_3: Value of $z_{i,j}$
- f_4: Average of all z values in neighborhood
- f_5: Normal to best fit plane of points in neighborhood
- f_6: Residual sum of squares (RSS) of best fit plane of points in neighborhood
- f_7: Pyramid filter
- f_8: Ray tracing score

These features are described in detail below.

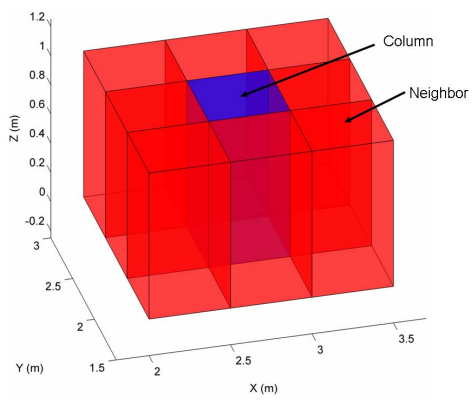

Fig. 2. A column and its neighbors.

f_1: Number of occupied columns in neighborhood

The first feature, f_1, is used to quantify the density of points around column (i,j). This feature encodes the fact that bushes, shrubs and tree trunks typically cast "shadows" in LIDAR data, thus reducing the number of occupied neighbors. As such, not every column (i,j) will contain data points, so there will be no minimum in these columns. To represent this, feature f_1 is the number of occupied columns, N, in the neighborhood of column (i,j):

$$f_1 = N \tag{1}$$

f_2, f_3, f_4: Features using z height values

Feature f_2 encodes the smoothness of the terrain around column (i,j). Intuitively, ground is expected to have a relatively smooth surface compared to the edges of trees or shrubs, which are expected to display larger jumps in height. To describe this, feature f_2 takes the difference between $z_{i,j}$ and the minimum z of all neighboring columns:

$$f_2 = \min\left(z_{i-1,j-1}, z_{i-1,j}, z_{i-1,j+1}, z_{i,j-1}, z_{i,j+1}, z_{i+1,j-1}, z_{i+1,j}, z_{i+1,j+1}\right) - z_{i,j} \tag{2}$$

Features f_3 and f_4 utilize the raw z value in each column, enforcing the assumption that the ground will not be located significantly higher than the LIDAR sensor (i.e. the ground probably won't be located at tree canopy height). This assumption is expected to be true except for cases with highly sloped terrain. Feature f_3 is taken as the value of $z_{i,j}$, and f_4 is defined as the average of all z values in the neighborhood of $z_{i,j}$:

$$f_3 = z_{i,j} \tag{3}$$

$$f_4 = \left(z_{i-1,j-1} + z_{i-1,j} + z_{i-1,j+1} + z_{i,j-1} + z_{i,j} + z_{i,j+1} + z_{i+1,j-1} + z_{i+1,j} + z_{i+1,j+1}\right)/N \tag{4}$$

f_5, f_6: Features using best fit plane

For features f_5 and f_6 a best fit plane is calculated for all points in the neighborhood of column (i,j). The plane minimizes orthogonal distances using principal component analysis. Given the set of N points in the neighborhood, where $\{P_k\}=\{[x_{i,j},\ y_{i,j},\ z_{i,j}]\}$, and the mean value $\overline{P} = (1/N)\sum_{k=1}^{N} P_k$, the normal vector n is calculated as

$$n = \underset{n\in\Re^3,\|n\|^2=1}{\arg\min} \sum_{k=1}^{N} \left(\left(P_k - \overline{P}\right)\cdot n\right)^2 . \quad (5)$$

Feature f_5 is the z-component of the normal vector, n:

$$f_5 = n\cdot(0,0,1) \quad (6)$$

The z-component is expected to be large for flat ground. This assumes that the ground is relatively flat. In sloped terrain, this feature can still be useful in discriminating between sloped terrain and near-vertical structures, such as tree trunk and rock boundaries.

As another means of measuring the smoothness of the terrain, feature f_6 is the normalized orthogonal RSS of the best fit plane:

$$f_6 = \frac{1}{N}\sum_{k=1}^{N} \left(\left(P_k - \overline{P}\right)\cdot n\right)^2 . \quad (7)$$

f_7: Pyramid filter

A ground filter similar to the one in [7] was also implemented. To improve computational efficiency, this filter is discretized by forming a pyramid structure (instead of a cone as in [7]) of cubic voxels under each point, $P_{i,j}$. The number of other minimum z points that fall within the pyramid of $P_{i,j}$ are counted, and this count is used as feature f_7. A representative pyramid of voxels is shown in Fig. 3.

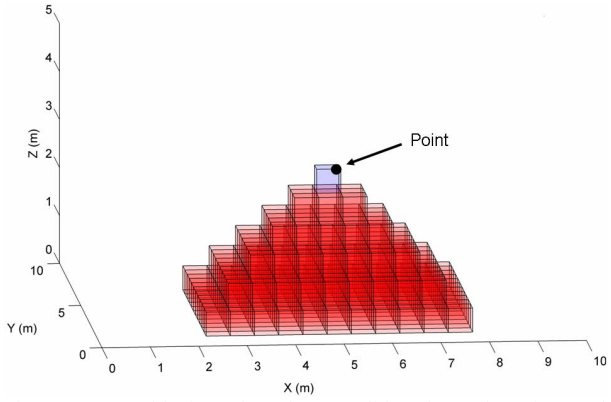

Fig. 3. A pyramid of voxels under a candidate data point. The voxel that contains the point is at the pyramid apex.

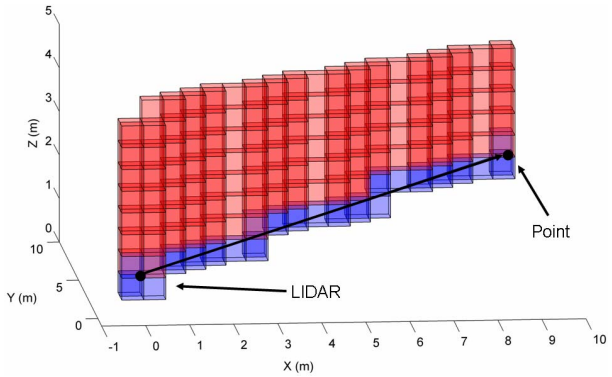

Fig. 4. Ray traced from data point back to LIDAR sensor source. Voxels above the traced line segment are red, and voxels along the line segment are blue. (Image best viewed in color.)

f_8: Ray Tracing Score

The last feature is inspired by ray tracing. Intuitively, it is obvious that the ground (or any other structure) cannot lie directly between the LIDAR and any point it observes. Similarly, the ground cannot lie above the line segment formed between the LIDAR and the points it observes. Feature f_8 quantifies this insight using a voxel-based approach. For each 0.5-m by 0.5-m by 0.5-m cubic voxel containing a point $P_{i,j}$, f_8 is a count of the line segments passing directly under that voxel. Thus, points with lower ray tracing scores are more likely to be ground, and points with higher scores are less likely. This concept is illustrated in Fig. 4.

Classifier Description

Given the feature vectors $F_{i,j} = (f_1,\ldots,f_8)$ for all of the points in a training scene and the hand-identified labels of whether each point is ground, a SVM classifier is trained. Based on cross-validation within the training data, a set of appropriate kernel parameters can be found. (For this work, the SVM classifier was implemented using LIBSVM [12], and based on cross-validation, a linear kernel was used with SVM parameter C=100.) Using the trained classifier, points belonging to a previously unlabeled scene can be classified.

III. Experiment Description

A. Equipment

For collection of LIDAR data, a nodding device was constructed using a camera tripod with a sensor platform mounted on top. The sensors include a small LIDAR sensor (Hokuyo UTM-30LX scanning range finder), and a MicroStrain 3DM-GX2 inertial measurement unit (IMU) which is used for inertially referencing the data. Fig. 5 shows a picture of this setup.

B. Environment

LIDAR data was acquired for different types of vegetated environments in the summer of 2009. The first set of data, *sparse,* was collected in Killian Court on the campus of MIT.

Fig. 5. Nodding device for collection of LIDAR data.

This data is meant to be a relatively simple baseline which will be compared to the other more complex scenes. A panoramic photo of this scene is shown in Fig. 6(a).

Two other data sets were collected from an arboretum located in the greater Boston area. Panoramic images of these environments are shown in Fig. 6(b) and Fig. 6(c). The *moderate* scene in Fig. 6(b) is relatively sparse. It contains several deciduous trees and shrubs, but is largely open. The *dense* scene, shown in Fig. 6(c), is more densely cluttered with numerous deciduous trees and shrubs.

C. Data Processing

To quantify the accuracy of ground classification, the LIDAR data collected for the scenes in Fig. 6 was hand-labeled. Each data point was classified into one of the four following categories: ground, bushes/shrubs, tree trunks, or tree branches/leaves.

Hand-labeling was done using Quick Terrain Modeler software from Applied Imagery [13]. Fig. 7 shows the classified LIDAR data projected into a global reference frame. Points are colored black, blue, red and green, representing ground, bushes/shrubs, tree trunks and tree branches/leaves, respectively. The LIDAR has 270 degree range with resolution of 0.25 degrees. The *sparse* scene only utilizes 180 degrees of this range, while the other sets use the full 270 degrees. Each scene has 200 scans, with LIDAR pitch angles ranging between –75 degrees and +55 degrees. So, the sparse scene has 144,400 data points and the other sets have 216,400 data points. In each scene, the axis of the nodding device was 0.55-m from the ground. Matlab was used for the computation of all feature values.

IV. Experimental Results

The two-stage approach for identifying points on the ground plane was applied to the experimental data sets, and the results are presented in Tables I-III. For the results in Table I, the *sparse* scene was used as the classifier test set, and the *moderate* and *dense* scenes were used as the classifier training set. For the results in Table II, the *moderate* scene was used as the test set, and the other two scenes were used as the training sets. Finally, in Table III, the *dense* scene was used as the test set, and the other two were used as the training sets.

The first two columns in each table show the results of stage 1, the local minimum z filter. Here it can be seen that this filter drastically reduces the number of points to be


(a)


(b)


(c)

Fig. 6. Panoramic images of (a) *sparse* scene, (b) *moderate* scene, and (c) *dense* scene.

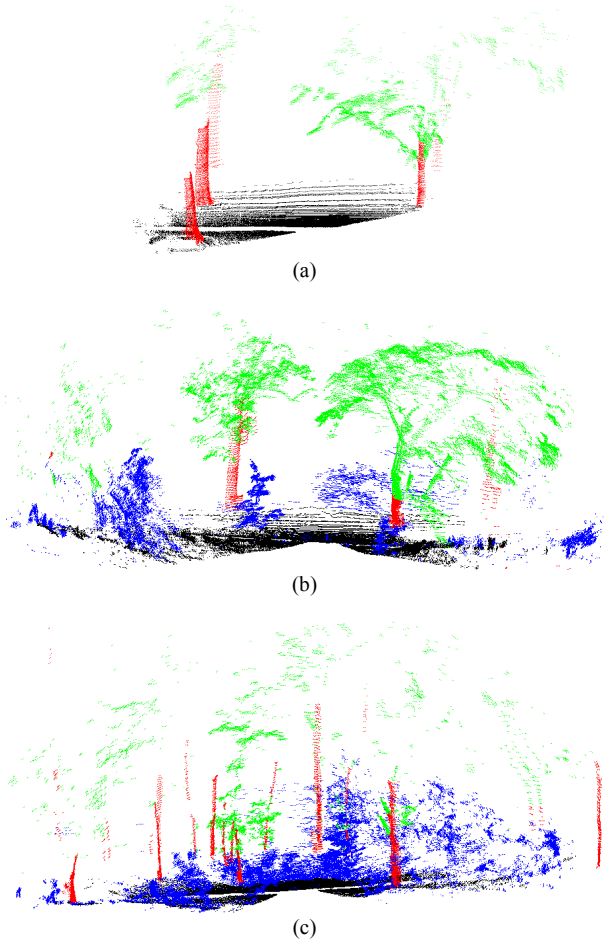

Fig. 7. Hand-labeled point clouds of (a) *sparse* scene, (b) *moderate* scene, and (c) *dense* scene. Black points indicate ground, blue points indicate bushes/shrubs, red points indicate trunks, and green points indicate canopy. (Image best viewed in color.)

analyzed. (Recall that a column might contain many ground points, but will contain at most one minimum z point.) It should be noted that the lower fraction of ground points that pass through the filter (as opposed to non-ground points) is due to the fact that the ground near the LIDAR is scanned much more densely than the trees and shrubs in the distance. In columns with both ground points and non-ground points, the first filtering stage selected ground points 100% of the time in the *sparse* data set and 98.4% of the time for the other two data sets.

The second two columns in each table show the results of stage 2, the SVM classifier, which identifies each of the minimum z points as being ground or non-ground. (The ground truth is the result of the hand-labeling described in Section III.C.) Here, it can be seen that the classifier can effectively identify points likely to belong to the ground class. The classification accuracy of the *sparse* scene was 91.12%. The classification accuracy for the *moderate* scene was 86.42%, and the accuracy for the *dense* scene was 78.09%. Here, accuracy is calculated as the number of points correctly classified divided by the total number of points classified. Bushes, trunks, branches, and leaves classified as

TABLE I
CLASSIFICATION RESULTS USING *SPARSE* SCENE AS TEST SET, WITH *MODERATE* AND *DENSE* SCENES AS TRAINING SETS

	Stage 1 Minimum z filter		Stage 2 SVM Classifier	
Ground Truth	Sensed points	Min z points	Ground	Non-Ground
Ground	53488	677	598	79
Bushes/ Shrubs	0	0	0	0
Tree Trunks	8324	26	4	22
Tree Branches /Leaves	7786	232	0	232

TABLE II
CLASSIFICATION RESULTS USING *MODERATE* SCENE AS TEST SET, WITH *SPARSE* AND *DENSE* SCENES AS TRAINING SETS

	Stage 1 Minimum z filter		Stage 2 SVM Classifier	
Ground Truth	Sensed points	Min z points	Ground	Non-Ground
Ground	74161	1131	928	203
Bushes/ Shrubs	30987	463	103	360
Tree Trunks	4483	47	7	40
Tree Branches /Leaves	32154	664	0	664

TABLE III
CLASSIFICATION RESULTS USING *DENSE* SCENE AS TEST SET, WITH *SPARSE* AND *MODERATE* SCENES AS TRAINING SETS

	Stage 1 Minimum z filter		Stage 2 SVM Classifier	
Ground Truth	Sensed points	Min z points	Ground	Non-Ground
Ground	65192	444	440	4
Bushes/ Shrubs	67403	752	340	412
Tree Trunks	11494	50	11	39
Tree Branches /Leaves	10314	379	1	378

"non-ground" are considered to be classified correctly.

Additionally, if a lower rate of false positives is desired (i.e. instances of bushes, trunks, branches, and leaves being incorrectly classified as ground), the threshold of the SVM classification can be adjusted. Fig. 8 shows the receiver operating characteristic (ROC) curves for the three data sets. For the solid line, the *sparse* scene was used as the classifier test set and the *moderate* and *dense* scenes were used as the classifier training sets. For the dashed line, the *moderate* scene was used as the test set, and the other two scenes were used as the training sets. Finally, for the dotted line, the *dense* scene was used as the test set, and the other two were used as the training sets.

Here it can be seen that low false positive rates can be achieved while maintaining relatively high true positive rates. For the *sparse* scene, a 70% true positive rate can be achieved with less than a 0.5% false positive rate. For the *moderate* scene, a 70% true positive rate can be achieved

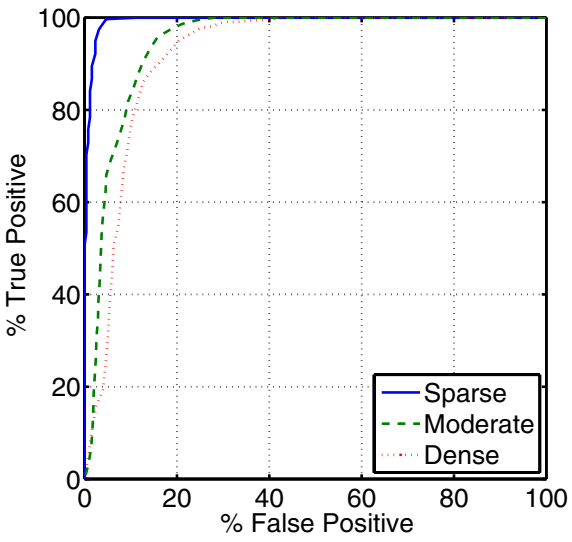

Fig. 8. ROC curves for *sparse* (solid), *moderate* (dashed) and dense *scene* (dotted)

with less than a 7% false positive rate. For the *dense* scene, a 70% true positive rate can be achieved with less than a 10% false positive rate.

All computations were performed using Matlab on a desktop computer with an Intel Core 2 Quad CPU at 2.40GHz and 3.25GB of RAM. Table IV shows average processing times for various parts of the computation. It should be noted that this computation can be easily parallelized and implemented in a faster programming language (e.g. C) if faster classification rates are desired.

TABLE IV
AVERAGE COMPUTATION TIMES FOR FEATURE EXTRACTION AND CLASSIFICATION

	Time/pt (s/pt)	Time/voxel (s/voxel)
$f_1 - f_6$	1.55×10^{-3}	1.66×10^{-4}
f_7 (cone filter)	3.86×10^{-3}	4.38×10^{-4}
f_8 (ray tracing)	1.22×10^{-4}	2.09×10^{-5}
SVM	4.04×10^{-6}	N/A

V. CONCLUSION

This paper has presented an approach for identifying ground points in 3-D LIDAR point clouds. This approach uses two stages. In the first stage, a local height-based filter eliminates a large percentage of the non-ground points. In the second stage, a classifier operating on eight geometrically defined features identifies which of the remaining points belong to the ground. The proposed approach was experimentally validated using LIDAR data collected from two forested environments. Results demonstrate that this approach can effectively discriminate between ground and non-ground points in such environments.

Future work will include validation of this approach using experimental data from geographically diverse environments and sloped terrain, optimizations of the underlying algorithms to decrease computation time, and use of the ground plane estimation to inform classification of other features in the environment. Furthermore, this work can be extended for use on a moving vehicle. This ground plane estimation could be the first step in the process of path planning for a UGV in a forested environment. Of course the accuracy of the final result would depend on the accuracy of mapping the data from the moving LIDAR into a stationary reference frame, which is a potential area for future research.

REFERENCES

[1] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, P. Mahoney, "Stanley: The robot that won the DARPA Grand Challenge," *Journal of Field Robotics*, 23(9), pp. 661-692, 2006.

[2] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, D. Ferguson, "Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics*, 25(8), pp. 425-466, 2008.

[3] P. Axelsson, "Processing of laser scanner data - algorithms and applications," *ISPRS Journal of Photogrammetry & Remote Sensing*, 1998.

[4] M. Elmqvist, "Ground surface estimation from airborne laser scanner data using active shape models," *International Archives of Photogrammetry and Remote Sensing*, Vol. XXXIV, Part 3A, pp. 114–118, 2002.

[5] K. Konolige, M. Agrawal, M. R. Blas, R. C. Bolles, B. P. Gerkey, J. Solà, A. Sundaresan, "Mapping, Navigation, and Learning for Off-Road Traversal," *Journal of Field Robotics*, 26(1), pp. 88-113, 2009.

[6] M. Hebert, N. Vandapel, "Terrain Classification Techniques From Ladar Data For Autonomous Navigation," *Collaborative Technology Alliances conference*, May 2003.

[7] J.-F. Lalonde, N. Vandapel, D. Huber, M. Hebert, "Natural Terrain Classification using Three-Dimensional Ladar Data for Ground Robot Mobility," *Journal of Field Robotics*, 23(10), 2006.

[8] C. Wellington, A. Courville, A. Stentz, "Interacting Markov Random Fields for Simultaneous Terrain Modeling and Obstacle Detection," *Proceedings of Robotics: Science and Systems*, June 2005.

[9] C. Wellington, A. Stentz, "Learning Predictions of the Load-Bearing Surface for Autonomous Rough-Terrain Navigation in Vegetation," *Field and Service Robotics Conference (FSR)*, 2003.

[10] Y.-H. Tseng, M. Wang, "Automatic Plane Extraction from LIDAR Data Based on Octree Splitting and Merging Segmentation," *Geoscience and Remote Sensing Symposium, IGARSS '05 Proceedings*, 2005.

[11] M. A. Fischler, R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the ACM*, 24(6), pp. 381-395, June 1981.

[12] C. Chang and C. Lin, "LIBSVM: a Library for Support Vector Machines" [Computer software], Oct. 2008. Available http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[13] Quick Terrain Modeler, Applied Imagery. Available: http://appliedimagery.com/