Terrain Classification and Identification of Tree Stems Using Ground-Based Lidar

Matthew W. McDaniel [a], Takayuki Nishihata [a], Christopher A. Brooks [a,*], Phil Salesses [a,b], Karl Iagnemma [a]

[a] *Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA*

[b] *Engineer Research and Development Center, US Army, Alexandria, VA 22315, USA*

* Corresponding author.

E-mail addresses: mcdaniel@alum.mit.edu (M.W. McDaniel), takayuki_nishihata@komatsu.co.jp (T. Nishihata), cabrooks@alum.mit.edu (C.A. Brooks), salesses@mit.edu (P. Salesses), kdi@mit.edu (K. Iagnemma).

Abstract

To operate autonomously in forested terrain, unmanned ground vehicles (UGVs) must be able to identify the load-bearing surface of the terrain (i.e. the ground) and obstacles in the environment. To travel long distances, they must be able to track their position even

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **DEC 2012**	2. REPORT TYPE	3. DATES COVERED **00-00-2012 to 00-00-2012**

4. TITLE AND SUBTITLE **Terrain Classification and Identification of Tree Stems Using Ground-Based Lidar**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Massachusetts Institute of Technology,Department of Mechanical Engineering, ,Cambridge,MA,02139**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S)
	11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

Journal of Field Robotics, Vol. 29, No. 6, pp. 891-910, November/December, 2012.

14. ABSTRACT

To operate autonomously in forested terrain, unmanned ground vehicles (UGVs) must be able to identify the load-bearing surface of the terrain (i.e. the ground) and obstacles in the environment. To travel long distances, they must be able to track their position even when the forest canopy obstructs GPS signals, e.g. by tracking progress relative to tree stems. This paper presents a novel, robust approach for modeling the ground plane and tree stems in forests from a single viewpoint using a lightweight lidar scanner. Ground plane identification is implemented using a two-stage approach. The first stage, a local height-based filter, discards most non-ground points. The second stage, based on a support vector machine (SVM) classifier, identifies which of the remaining points belong to the ground. Main tree stems are modeled as cylinders or cones to estimate the diameter 130 cm above the ground plane. To fit these models, candidate main stem data is selected by finding points approximately 130 cm above the ground. These points are clustered into separate point clouds for each stem. Cylinders and cones are fit to each point cloud, and heuristic filters identify which fits correspond to tree stems. Experimental results from five forested environments demonstrate the effectiveness of this approach. For ground plane estimation, the overall classification accuracy was 86.28% with a mean error for the ground height of approximately 4.7 cm. For stem estimation, up to 50% of main stems were accurately modeled using cones, with a root mean square diameter error of 13.2 cm.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**49**	

when the forest canopy obstructs GPS signals, e.g. by tracking progress relative to tree stems.

This paper presents a novel, robust approach for modeling the ground plane and tree stems in forests from a single viewpoint using a lightweight lidar scanner. Ground plane identification is implemented using a two-stage approach. The first stage, a local height-based filter, discards most non-ground points. The second stage, based on a support vector machine (SVM) classifier, identifies which of the remaining points belong to the ground.

Main tree stems are modeled as cylinders or cones to estimate the diameter 130 cm above the ground plane. To fit these models, candidate main stem data is selected by finding points approximately 130 cm above the ground. These points are clustered into separate point clouds for each stem. Cylinders and cones are fit to each point cloud, and heuristic filters identify which fits correspond to tree stems.

Experimental results from five forested environments demonstrate the effectiveness of this approach. For ground plane estimation, the overall classification accuracy was 86.28% with a mean error for the ground height of approximately 4.7 cm. For stem estimation, up to 50% of main stems were accurately modeled using cones, with a root mean square diameter error of 13.2 cm.

Keywords

Forestry, terrestrial robotics, perception

1. Introduction

Classification and modeling of forested terrain is a major challenge in remote sensing. Accurate modeling of terrain is critical for autonomous navigation of unmanned ground vehicles (UGVs) and inventory management in the forestry industry. These applications require that a sensing system identify contours of the load-bearing surface (i.e. the ground) and recognize main tree stems. While vehicle-mounted terrestrial lidar sensors (TLS) have the ability to return high-density point clouds from below the forest canopy, extracting estimates of ground height and locations and diameters of tree stems is nontrivial.

In the autonomous mobility domain, quickly and accurately modeling the environment in the forest is an important step towards autonomous UGV operation. Main stem locations, collected from a UGV-mounted lidar, can be used as inputs for simultaneous localization and mapping (SLAM) algorithms (Pradalier & Sekhavat, 2003). Digital terrain models (DTMs) generated using vehicle-mounted sensors can also be correlated to aerial data to enhance UGV localization (Vandapel et al., 2006; Carle et al., 2010). In the forestry domain, tree stem modeling can be used for estimating biomass (Lucas et al., 2006; Sharma and Parton, 2009; Li and Weiskittel, 2010) or characterizing forest stand structure (Lefsky et al., 2005; Lefsky et al., 1999).

Ground plane estimation based on terrestrial lidar can be challenging due to the occlusions of terrain by tree trunks and low vegetation. Due to the low observation angle of TLS, as compared with aerial laser scanning (ALS), nearby terrain can also occlude more distant terrain. Previous researchers have proposed several approaches to address these challenges. Maas et al. (2008) combined sensor scans from multiple viewpoints to avoid occlusions, but this approach relies on accurately registered scans. Raber et al. (2002) identified different types of vegetation based on a vertical histogram, but relied on the consistent observation angle available in aerial lidar data. Working in an agricultural scenario, Wellington & Stentz (2003) and Wellington et al. (2005) assumed that for any given horizontal coordinate either the ground or the upper extent of the vegetation above it will have lidar returns, but this may not be true for terrestrial lidar in forests. Other researchers assumed explicit constraints on the terrain geometry, including height relative to the scanner (Aschoff et al., 2004) and maximum slope (Aschoff et al., 2004; Lalonde et al., 2006). Rather than using strict bounds on each geometric attribute, this paper combines height, slope, error from a plane fit, and other features using a novel machine learning approach to classify a lidar point as belonging to the ground surface.

Tree stem modeling in the forest based on terrestrial lidar can be challenging due to the presence of understory vegetation (e.g. shrubs), low branches, and multiple trees, which make it difficult to identify the lidar points corresponding to each tree. One promising approach, proposed in (Forsman & Halme, 2005), fits circles to segments of individual lidar scan lines, but no estimation of the accuracy of the algorithm is available. Other researchers have demonstrated measurement of tree stems diameters using lidar data (e.g. Henning & Radtke, 2006; Maas et al., 2008) with RMS errors of 1.0 cm, but this has been done with little or no understory vegetation present, making the tree stems clearly detectable. Even without understory vegetation, researchers have shown that automatic detection of trees is challenging, particularly when using lidar data from a single viewpoint (Thies & Spiecker, 2004). This paper proposes an approach to measuring stem locations and diameters from a single viewpoint, and presents results from five test environments, four of which include understory vegetation.

A flowchart for the proposed approach for ground plane identification and stem diameter estimation is shown in Fig. 1. As with previous work (Wellington & Stentz, 2003; Aschoff et al., 2004), ground plane estimation begins with a local height-based filter, selecting the lowest lidar return in each 0.5 m × 0.5 m horizontal grid cell.[1] Next, a set of eight geometrically defined features are computed for each of the remaining points. A support vector machine (SVM), trained on hand-labeled lidar data, operates on the eight computed features to identify points belonging to the ground.

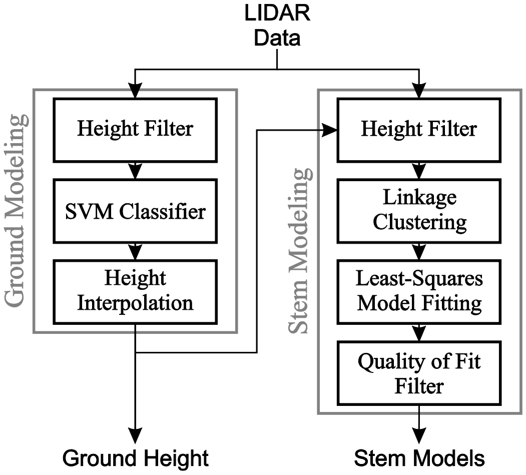


Fig. 1. Information flow in proposed ground plane estimation and stem modeling approach.

To model main tree stems, candidate stem data is selected by taking a horizontal slice of lidar data centered at 130 cm above the estimated ground height. This data is then clustered to determine which points belong to the same stems. Stems are modeled by fitting each cluster of data to cylinders and cones. Finally, physics-based constraints are introduced to assess the

[1] These dimensions were chosen to roughly correspond to the width of our robotic data collection platform, which has a width of approximately 0.5 m.

quality of these fits, and poor fits are discarded. The effectiveness of this approach is demonstrated using data collected from five different forested environments, and estimated stem diameters are compared against stem diameters measured using a diameter tape. Note that this approach is intended specifically to identify tree stems. General lidar-based obstacle detection algorithms have been described elsewhere, e.g. (Manduchi et al., 2005).

2. Methods

2.1 Study area

Study areas for this work included three different vegetated environments near Boston, Massachusetts, USA. One research location was the Arnold Arboretum of Harvard University. The total area of the arboretum is 265 acres and includes moderately sloped terrain. The Arnold Arboretum contains 4,099 taxa, with most species hailing from North America and Eastern Asia. Stands are mixed age and very diverse, making this an appealing test site in terms of tree variety.

Another research area was Breakheart Reservation. This reservation covers 640 acres and has highly sloped terrain. Stands are predominately oak, hemlock and pine.

A manicured landscape was also selected for initial testing, and was collected in Killian Court on the campus of the Massachusetts Institute of Technology. The terrain in Killian Court is flat and contains mature oak trees.

2.2 Lidar data

To facilitate the collection of lidar data, a nodding (i.e. automated tilt) device was constructed by mounting a sensor platform atop a camera tripod. Sensors included a Hokuyo UTM-30LX scanning range finder and a MicroStrain 3DM-GX2 inertial measurement unit

(IMU) for inertially referencing the Hokuyo data. The UTM-30LX lidar scans an angular range of 270° in 25 ms, with an angular resolution of 0.25° along the scan line, a depth range of 30 m, and an accuracy of ±50 mm. It weighs only 370 g and was selected because it is light enough to be mounted on a small man-portable mobile robot. For comparison, the high-resolution lidar scanners used in previous studies (Thies & Spiecker, 2004; Forsman & Halme, 2005; Maas et al., 2008) have weighed 14 kg or more. The orientation of the setup, as detected using the IMU, was recorded immediately following each lidar scan. A laptop was also used to stream data from the sensors. Fig. 2 depicts the setup. Note that while many TLS applications use a lidar scanner oriented so the scan lines lie in a vertical plane, here the plane of the scans is roughly horizontal. This allows a single scan to intersect multiple tree stems, which is expected to improve estimation of relative positions of tree stems in future studies using a moving platform.

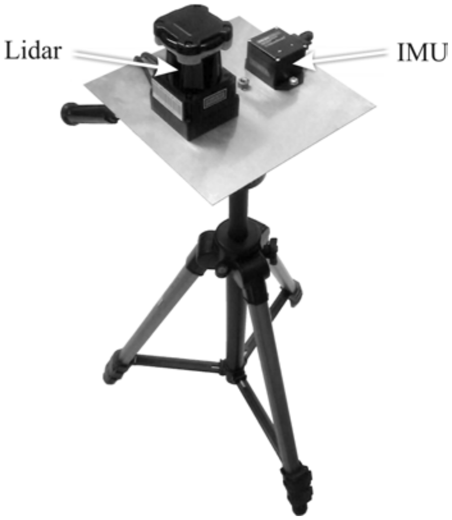


Fig. 2. Nodding device for collection of lidar data.

Three of the five data sets presented in this paper were collected in the summer of 2009. The first set of data, *baseline*, was collected in Killian Court on the campus of MIT. A panoramic photo of this scene is shown in Fig. 3(a) and contains several mature oak trees distributed about a near-flat ground plane. This data set, meant to serve as a baseline, will be compared to the other, more challenging scenes. Two other data sets, *sparse* and *moderate*, were collected at Harvard's Arnold Arboretum; panoramic images of these environments are shown in Fig. 3(b) and Fig. 3(c), respectively. The *sparse* scene in Fig. 3(b) contains several deciduous trees and shrubs, but is largely open. The *moderate* scene, shown in Fig. 3(c), is cluttered with numerous deciduous trees and shrubs, and significant ground cover.

The remaining two data sets, *dense1* and *dense2* were collected at Breakheart Reservation in January of 2010. Panoramic images of these scenes, *dense1* and *dense2,* are shown in Fig. 3(d) and Fig. 3(e), respectively. Both scenes contain a significant number of deciduous trees and have considerably sloped terrain. Since both scenes were collected during winter, there was snow on the ground and no leaf cover on the deciduous trees.

For each of these scenes, the nodding device was operated by hand, and swept between pitch angles of –85º and +55º, with scans spaced at 0.7° on average. The *baseline* scene utilized 180º of the angular range of the lidar, while the other sets used the full 270º. Each scene encompasses 200 scan lines and can contain as many as 216,200 data points (slight variation is caused when the nearest surface, within the lidar line-of-sight, is out of range).

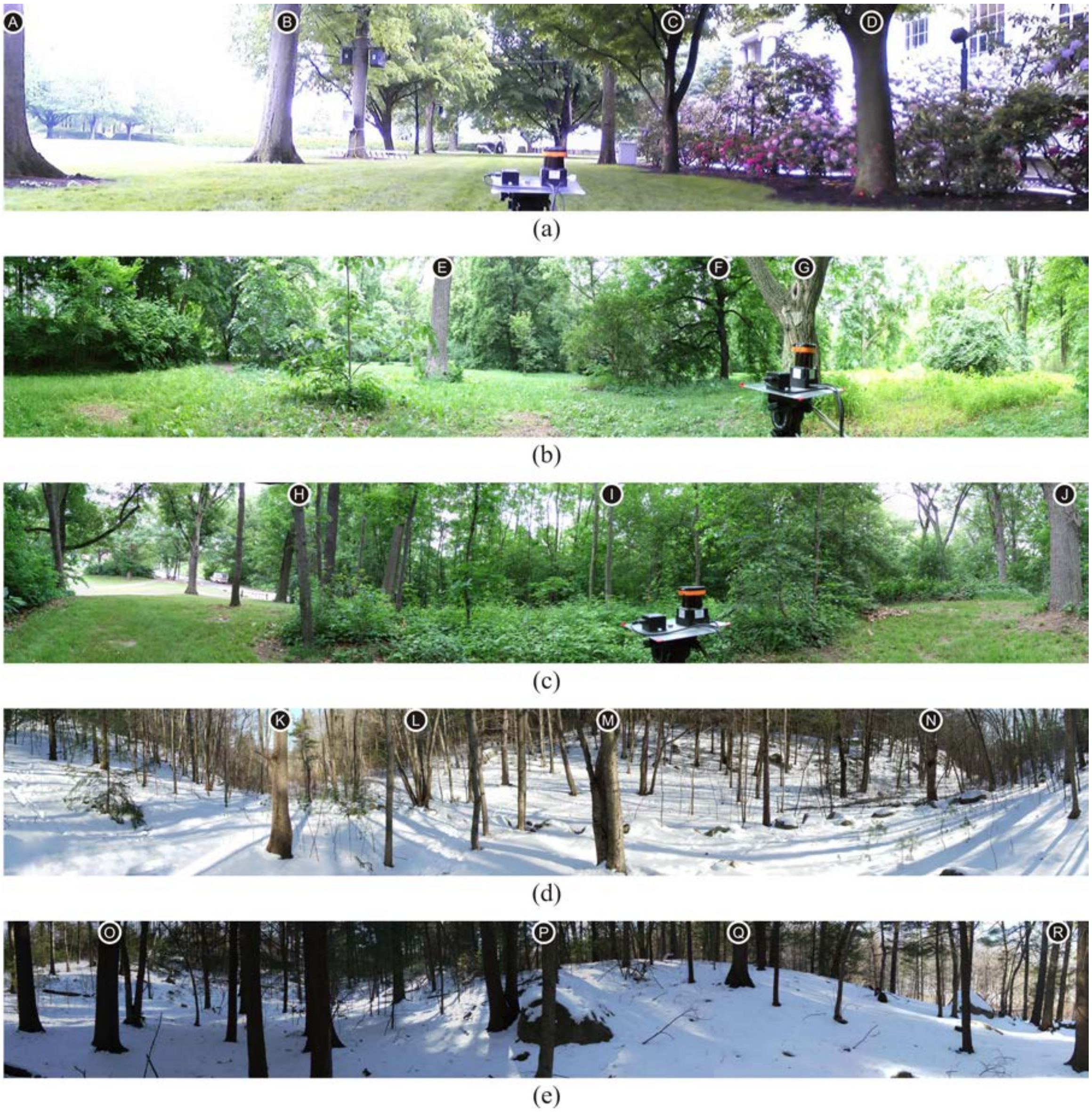


Fig. 3. Panoramic images of (a) *baseline* (b) *sparse*, (c) *moderate*, (d) *dense1* and (e) *dense2* scenes.

These five scenes were chosen with the intention of gathering diverse data to illustrate that the classification and modeling techniques in this paper can be applied to a wide range of forested environments, and not only carefully selected test cases. For example, when estimating the ground plane, it is important to test algorithm performance on sloped terrain, because slopes are generally more difficult to correctly estimate than flat terrain surfaces. This is a recognized challenge in the literature (Aschoff et al., 2004), where inadequate ground plane estimation is often obtained on sloped terrain. This issue is addressed by including hilly terrain in our testing. The *dense1* scene exhibits a total elevation change of 4.2 m over the 30 m range of the sensor, and lidar scanning is performed facing down a slope. The *dense2* scene exhibits a total change of 3.1 m over 30 m, and lidar scanning is performed facing up the hill. Table 1 shows the total elevation change for each of the five data sets as estimated through manual examination of the lidar point clouds.

Table 1

Statistics for Experimental Test Scenes

Scene	Total Elevation Change	Number of Trees
baseline	0.681 m	4
sparse	1.784 m	5
moderate	1.832 m	16
dense1	4.202 m	48
dense2	3.126 m	40

For robust stem estimation, a range of different tree sizes and types were included in the test data. The five experimental test scenes presented in this paper contained trees of various species and sizes, allowing for testing over a range of tree sizes and shapes. The *baseline* scene, collected at MIT in Killian Court, contained mature oak trees with tall, easily visible trunks. The *sparse* and *moderate* scenes from Arnold Arboretum, contained mostly broadleaved trees, some with large distinct trunks and others with low trunk-obscuring branches and leaves. Breakheart Reservation, where the *dense1* and *dense2* scenes were collected, contained a mixture of broadleaved and needle-leaved trees, including some sparse evergreen foliage. Among the five scenes, 113 trees were sensed. Table 1 shows the number of trees in each scene.

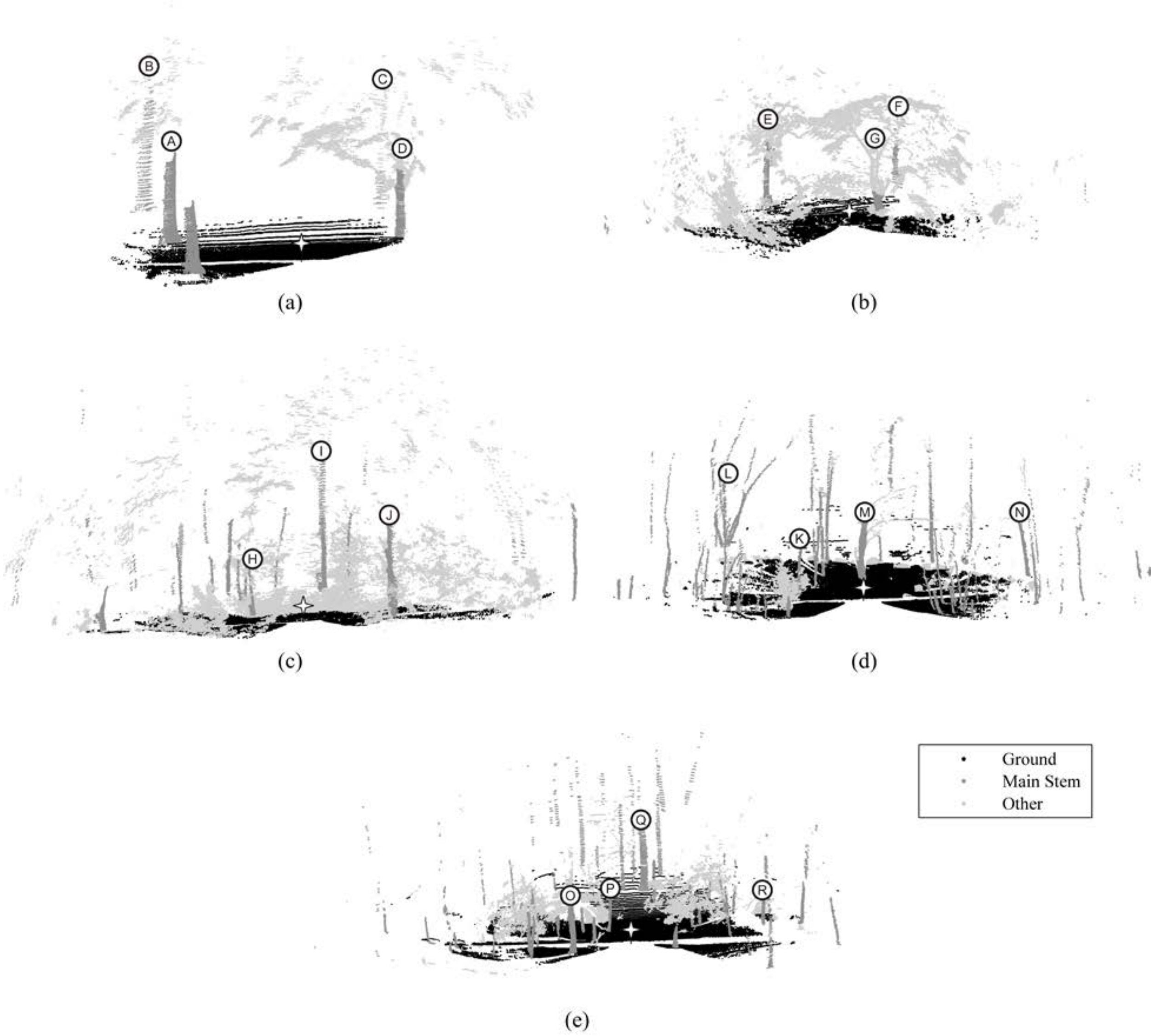
The scene selection also incorporated seasonal variation: the *baseline*, *sparse* and *moderate* scenes were collected during summer, and the *dense1* and *dense2* scenes were collected during winter. This affects ground plane estimation because a 6 cm layer of snow is present in the winter scenes. It also affects stem estimation because there are no leaves on deciduous trees after the seasonal abscission, reducing the number of non-stem lidar returns. A robust algorithm should yield consistent results across environments and seasons.

2.3 Field data

Stem diameter at breast height (1.3 m, dbh) was measured for each of the trees in the five scenes using a diameter tape. Field measured dbh values were obtained for comparison to diameters estimated by the lidar stem modeling algorithm. To identify which diameter corresponded with which tree, a rough map of the trees was created in the field. The X-Y position of each tree in the lidar reference frame was identified afterwards based on manual examination of the lidar point clouds.

2.4 Data Processing

To quantify the accuracy of the algorithms presented, the lidar data was hand-labeled using Quick Terrain Modeler, a software package developed by Applied Imagery (Quick Terrain Modeler, 2009). Each data point was classified into one of the following categories: (1) ground, (2) bushes/shrubs, (3) tree main stems and (4) tree canopy. For situations when the class of the lidar point was unclear, photographs of the environment (shown in Fig. 3) were used to assist in classification. For the work in this paper, the key distinction is between “ground” and “not ground,” and this information is used to train the classifier. Other classes are used only to get a fine-grained analysis of classification results. Fig. 4 shows the hand-labeled lidar data projected into a global reference frame, with the location of the lidar marked with a star and trees labeled with letters to show correspondence to Fig. 3.



(a)

(b)

(c)

(d)

(e)

Fig. 4. Hand-labeled point clouds of (a) *baseline* (b) *sparse*, (c) *moderate*, (d) *dense1* and (e) *dense2* scenes.

2.5 Ground Plane Estimation Algorithm

Given a set of lidar data points in Cartesian space, the goal of ground plane estimation is to identify which of those points belong to the ground. The approach proposed here divides the ground plane estimation task into two stages. The first stage is a local height-based filter, which encodes the fact that, in any vertical column, only the lowest points may belong to the ground. In practice, this eliminates a large percentage of non-ground points from further consideration. The second stage is a support vector machine (SVM) classifier, which combines eight geometric measures—the classification features—to determine which of the remaining points belong to the ground plane. All algorithms presented in this paper were implemented in Matlab.

2.5.1 Stage 1: Height Filter

Candidate points are represented in an inertial frame with coordinates (*x*,*y*,*z*), where the *x* and *y* axes lie on a horizontal plane, and the *z* axis is positive upward. The point cloud volume is divided/tessellated into 0.5 m × 0.5 m vertical columns identified by indices (*i*,*j*), where $i = \lfloor x/0.5 \rfloor$ and $j = \lfloor y/0.5 \rfloor$. Thus, a point located at (2.9 m, 4.1 m, 1.7 m) 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 as a vector, $\mathbf{P_{i,j}}$, with coordinates [$x_{i,j}$, $y_{i,j}$, $z_{i,j}$].

2.5.2 Stage 2: Feature Extraction

In the second stage of this approach, a variety of features are used to represent attributes of each lowest point $\mathbf{P_{i,j}}$ and the lowest points in each of the neighboring columns. Here, the neighborhood of a column (*i,j*) is defined as the column and its eight 8-connected neighboring columns.

A set of eight features was defined based on their usefulness in discriminating ground from non-ground. These features, denoted $f_1,\ldots,f_8$ are combined into a feature vector $\mathbf{F_{i,j}}=[f_1,\ldots,f_8]$ 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

The features above are described in detail below.

f_1: Number of occupied columns in neighborhood

f_1 is the number of occupied columns, *N*, in the neighborhood of column (*i,j*):

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

where an occupied column is defined as a column containing at least one point. This feature is used to quantify the density of points around column (*i,j*). This measures occlusion that bushes, shrubs and tree trunks typically create in lidar data, thus reducing the number of occupied neighbors.

f_2, f_3, f_4: Features using z height values

Feature f_2 calculates the difference of $z_{i,j}$ and the minimum z of all neighboring columns:

$$f_2 = \min(z_{i-1,j-1}, z_{i-1,j}, z_{i-1,j+1}, z_{i,j-1}, z_{i,j+1}, z_{i+1,j-1}, z_{i+1,j}, z_{i+1,j+1}) - z_{i,j}\,. \quad (2)$$

This feature encodes the unevenness of terrain around column (i,j). Intuitively, ground is expected to have a relatively smooth surface compared to the edges of tree stems or shrubs. In the case of smooth ground, f_2 will be near zero. When $\mathbf{P_{i,j}}$ is part of a tree or shrub and has a neighboring point on the ground, f_2 will be relatively large.

f_3 is defined as the value of $z_{i,j}$, and f_4 is the average of all z values in the neighborhood of $z_{i,j}$:

$$f_3 = z_{i,j} - z_{lidar} \quad (3)$$

$$f_4 = (z_{i-1,j-1}+z_{i-1,j}+z_{i-1,j+1}+z_{i,j-1}+z_{i,j}+z_{i,j+1}+z_{i+1,j-1}+z_{i+1,j}+z_{i+1,j+1}) \,/\, f_1 - z_{lidar}, \quad (4)$$

where z_{lidar} is the height of the lidar sensor. Features f_3 and f_4 utilize the z value in each column. This allows the classifier to consider that points significantly higher than the lidar sensor are less likely to be ground than points lower 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 extremely sloped terrain.

f_5, f_6: Features using best fit plane

For features f_5 and f_6, a best fit plane is calculated using all points in the neighborhood of column (i,j). Given the set of N points in the neighborhood of column (i,j), where each point is denoted $\mathbf{P_k}$, the mean is computed as:

$$\overline{\mathbf{P}} = \frac{1}{N}\sum_{k=1}^{N}\mathbf{P_k} \quad (5)$$

The normal vector of the best-fit plane that minimizes orthogonal distances to the points in the neighborhood of column (*i*,*j*) is calculated as:

$$\mathbf{n_{i,j}} = \underset{\mathbf{n}\in\Re^3, \|\mathbf{n}\|^2=1}{\arg\min} \sum_{k=1}^{N} \left(\left(\mathbf{P_k} - \overline{\mathbf{P}} \right) \cdot \mathbf{n} \right)^2 \tag{6}$$

Feature f_5 is the *z*-component of the unit normal vector, $\mathbf{n_{i,j}}$:

$$f_5 = \mathbf{n_{i,j}} \cdot (0,0,1). \tag{7}$$

f_5 is expected to have a value near one for flat horizontal ground. This is useful for differentiating between the ground and near-vertical structures, such as steep rock faces.

Feature f_6 is the normalized orthogonal residual sum of squares (RSS) of the best fit plane:

$$f_6 = \frac{1}{N} \sum_{k=1}^{N} \left(\left(\mathbf{P_k} - \overline{\mathbf{P}} \right) \cdot \mathbf{n_{i,j}} \right)^2 \tag{8}$$

This feature measures smoothness of the terrain around column (*i*,*j*). The ground is expected to lie nearly within a plane. Meanwhile, discontinuities will occur at the edges of trees, shrubs, or large rocks, and this will be reflected in a larger RSS of the best fit plane. Although this feature will be similar to f_2 on open flat ground, it is intended to provide additional robustness to ground point classification on sloped terrain.

f_7: Pyramid filter

An inverted pyramid filter was used as feature f_7, similar to the inverted cone filter used in Thies et al. (2004). Here, feature f_7 is computed as the number of other minimum *z* points which lie within a pyramid of 0.5-meter cubic voxels with its apex at point $\mathbf{P_{i,j}}$. This discretized pyramid approach was used to improve computational efficiency over the inverted cone filter.

f_8: Ray Tracing Score

The last feature involves ray tracing. 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 any point it observes. Feature f_8 quantifies this insight using a voxel-based approach. For each 0.5 m × 0.5 m × 0.5 m cubic voxel containing a lowest point, $\mathbf{P_{i,j}}$, f_8 is a sum of the number of line segments that pass through a voxel in column (i,j) that is below the voxel that contains $\mathbf{P_{i,j}}$. Intuitively, a point lying on flat ground should have a ray tracing score of zero. A point lying on sloped ground should be close to zero. The ray tracing concept is illustrated in Fig. 5. The heavy line represents a ray traced from the lidar to a data point. The voxels that this ray passes through are shaded, and the voxels above this ray are outlined but transparent. Any $\mathbf{P_{i,j}}$ in a transparent voxel would have its ray tracing score incremented by one.

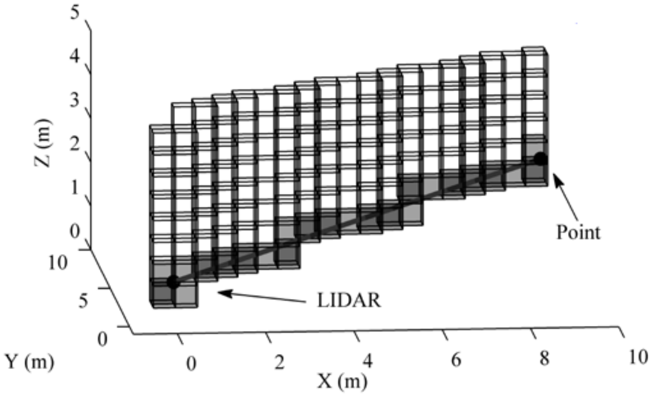


Fig. 5. Ray tracing visualization

This process can be represented mathematically such that the ray tracing score for a point $P_{i,j}$ is the count of points $P_{i1,j1}$ located such that their line segments pass below the voxel containing $P_{i,j}$. For example $P_{i,j}$, with $i > \lfloor x_{lidar} / 0.5 \rfloor$, $j > \lfloor y_{lidar} / 0.5 \rfloor$ and $\lfloor z_{i,j} / 0.5 \rfloor < \lfloor z_{lidar} / 0.5 \rfloor$:

$$
\begin{aligned}
& f_8\left(\mathbf{P}_{i,j}\right) = count\left(\mathbf{P}_{i1,j1}\right) \quad \forall\, i_1,\, j_1 \quad \text{such that} \\
& \quad i_1 \geq i, \quad j_1 \geq j, \\
& \quad (x_{i1,j1} - x_{lidar})(0.5 \cdot (j+1) - y_{lidar}) > (y_{i1,j1} - y_{lidar})(0.5 \cdot i - x_{lidar}) \\
& \quad (x_{i1,j1} - x_{lidar})(0.5 \cdot j - y_{lidar}) < (y_{i1,j1} - y_{lidar})(0.5 \cdot (i+1) - x_{lidar}) \\
& \quad (y_{i1,j1} - y_{lidar})(0.5 \cdot (\lfloor z_{i,j} / 0.5 \rfloor + 1) - z_{lidar}) < (z_{i1,j1} - z_{lidar})(0.5 \cdot (j+1) - y_{lidar}), \text{ and} \\
& \quad (z_{i1,j1} - z_{lidar})(0.5 \cdot (i+1) - x_{lidar}) > (x_{i1,j1} - x_{lidar})(0.5 \cdot (\lfloor z_{i,j} / 0.5 \rfloor + 1) - z_{lidar})
\end{aligned} \quad . \tag{9}
$$

Similar equations can be written for other ranges of *i*, *j*, and $z_{i,j}$.

2.5.3 Stage 2: SVM Classifier Description

Given the feature vectors $\mathbf{F}_{\mathbf{i,j}}$=[$f_1$,…,$f_8$] for all potential ground points $\mathbf{P}_{\mathbf{i,j}}$ in a training set, and the hand-attributed labels of whether each point $\mathbf{P}_{\mathbf{i,j}}$ lies on the ground surface, a SVM classifier is trained (Schölkopf, 2000). Tunable kernel parameters, including the kernel and misclassification cost *C* were found by minimizing the cross-validation error rate within the training data. For this work, the SVM classifier was implemented using LIBSVM (Chang & Lin, 2008), and based on cross-validation, a linear kernel was used with misclassification cost *C*=100. Using the trained classifier, minimum *z* points belonging to a previously unlabeled scene can be classified as ground or not ground.

2.5.4 Ground Plane Modeling

If a column contains a ground point from the classifier stage, the height of that point is used as the ground height in that column. In a column not containing a classified ground point, such as when occlusions or sparse data leave gaps between ground points, interpolation is necessary to estimate the ground height. For this work, Delaunay triangulation is used to interpolate between classified ground points. Thus, the ground height for a column is calculated as the height of the Delaunay triangle at the center of the column.

2.6 Stem Estimation Algorithm

The proposed approach models the main stems of trees (i.e. trunks) using cylinders and cones as geometric primitives. The first step of this process takes a horizontal slice of lidar data vertically centered at breast height. These points are then clustered, and fit to cylinders and cones. Finally, physical constraints are used to reject unrealistic fits.

2.6.1 Breast-Height Slice

The first step of stem estimation is to identify which lidar data points belong to a single main stem. In the approach described here, candidate main stem points are chosen based on their height above the ground. This is based on the intuition that at a certain height, stem data is likely to be above low lying vegetation and obstructions (e.g. shrubs, grass, rocks) and below higher vegetation (e.g. secondary stems and other canopy). For this work, we used the standard 130 cm breast height.

To obtain sufficient data to accurately model tree stems, candidate lidar data must be gathered within a pre-specified vertical range above and below breast height. The size of this range is important because it has significant effects on tree modeling results. A larger range typically contains more lidar data, which generally leads to more accurate cylinder and cone fits, while a smaller range often reduces unwanted non-trunk data from above (canopy) and below (shrubs, rocks, etc.). However, if the range is too small it could miss a tree entirely, due to occlusion or data sparsity. Here, the range is selected to balance these two competing effects.

In the literature (Aschoff et al., 2004; Henning & Radtke, 2006), very thin slices of data have been used (10 cm and roughly 2 cm, respectively), enabled by very high resolution lidar sensors and relatively close standoff distance. Generally, the slice width should be large enough to contain lidar data at the working range of the sensor. This means that the width must be greater than the vertical distance between two scan lines at maximum working range (see Fig. 6). The Hokuyo UTM-30LX lidar used for this work has a functional range of 29 m and the average change in pitch angle during collection was 0.7º. At a range of 29 m, the difference in height of two consecutive horizontal scans is approximately 35 cm, so the slice width must be at least 35 cm to ensure that the slice contains at least one horizontal scan.

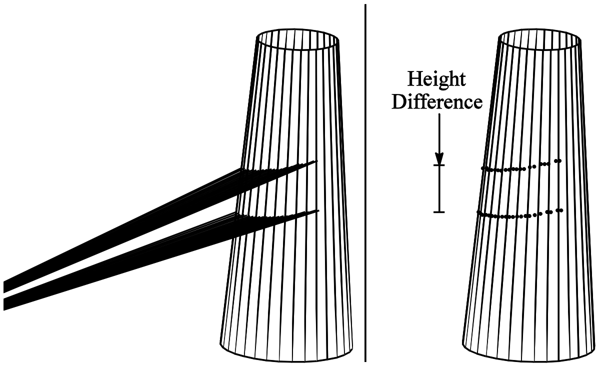


Fig. 6. Two consecutive vertical lidar scan lines, and height difference.

For this work, a two step strategy was implemented to obtain candidate main stem data. The first step identifies probable stem locations in the X-Y plane while the second step selects a set of data points at each of those locations to be used for model fitting. Since the ground plane estimation algorithm in this paper uses a 0.5 m × 0.5 m grid cell discretization, this algorithm will be discretized at the same resolution.

Barring occlusion, every tree taller than 1.3 m should have trunk data at a height of 130 cm above the ground.[2] In addition, 130 cm was observed to be higher most of the non-tree

[2] It should be noted that trees with dbh less than 13 cm may also be missed at a range of 29 m, due to the horizontal spacing of points between scans.

vegetation, while still being below most of the trees' foliage. So, to identify stem locations, a search is performed at this height. Since the difference in the height of two consecutive scans can be as large as 38 cm, a range of 40 cm was used for this algorithm (i.e. 130 ± 20 cm). If a column contains any lidar data within this 40 cm range, that column is presumed to contain trunk data. For each of these columns, a larger 60 cm slice of data (i.e. 130 ± 30 cm) is then selected as candidate main stem data to be used to model fitting. All data points within that column and range will be selected for analysis. This slice width of 60 cm was chosen based on empirical data from the five test scenes. This analysis is presented in Section 3.2.3..

To determine whether a lidar data point (x,y,z) lies within the 60 cm slice width, the height of that point is calculated with respect to the ground height described in Section 2.5.4. If a data point is in a column that does not have an estimation for ground height (i.e. it lies outside the convex hull of the convex hull of the sensed ground points), it will not be considered a candidate main stem point regardless of its height.

2.6.2 Clustering

The next step in the stem estimation algorithm is to determine which candidate main stem data points belong to individual trees, since one stem could have points in multiple columns. This is accomplished by clustering the candidate data points. The most prevalent clustering algorithm is K-Means, which assigns data points to clusters such that each point belongs to the cluster with the nearest centroid. This method requires that the number of clusters is known *a priori*, however the number of trees in any given forested scene is generally unknown.

Instead of K-Means, a single-linkage clustering algorithm is applied in this work (Jain & Dubes, 1988). In this algorithm, any two points which are closer to each other than a pre-specified threshold distance *l* are considered to be “connected” and belong to the same cluster. Here *l* is 50 cm. Two points are considered to be part of different clusters if there is no chain of such connected points linking the same cluster. For this work, the threshold physically implies that the distance between the visible surfaces of any two neighboring trees cannot be less than *l*.

Single-linkage clustering algorithms typically require that the distances between all pairs of points be calculated, a process that is computationally costly when (as in the *dense1* and *dense2* datasets) there are over 10,000 data points to be clustered. To address this problem a two-stage clustering approach is used, as illustrated in Fig. 7. To cluster the points shown in Fig. 7(a), the first step is to calculate the occupied grid cells (Fig. 7(b)). These grid cells are then clustered, such that any occupied grid cells within 50 cm of each other belong two the same cluster (Fig. 7(c)). Finally, for each grid cell cluster (e.g. “Cluster B”), the points within that cluster are themselves clustered with a threshold distance of 50 cm (e.g. into “Cluster B1” and “Cluster B2”). Each of these final point clusters (“A1,” “B1,” “B2,” and “C1”) is assumed to correspond to a single tree. For the data sets used in this paper, this two-stage approach yielded a 12-fold increase in clustering speed.

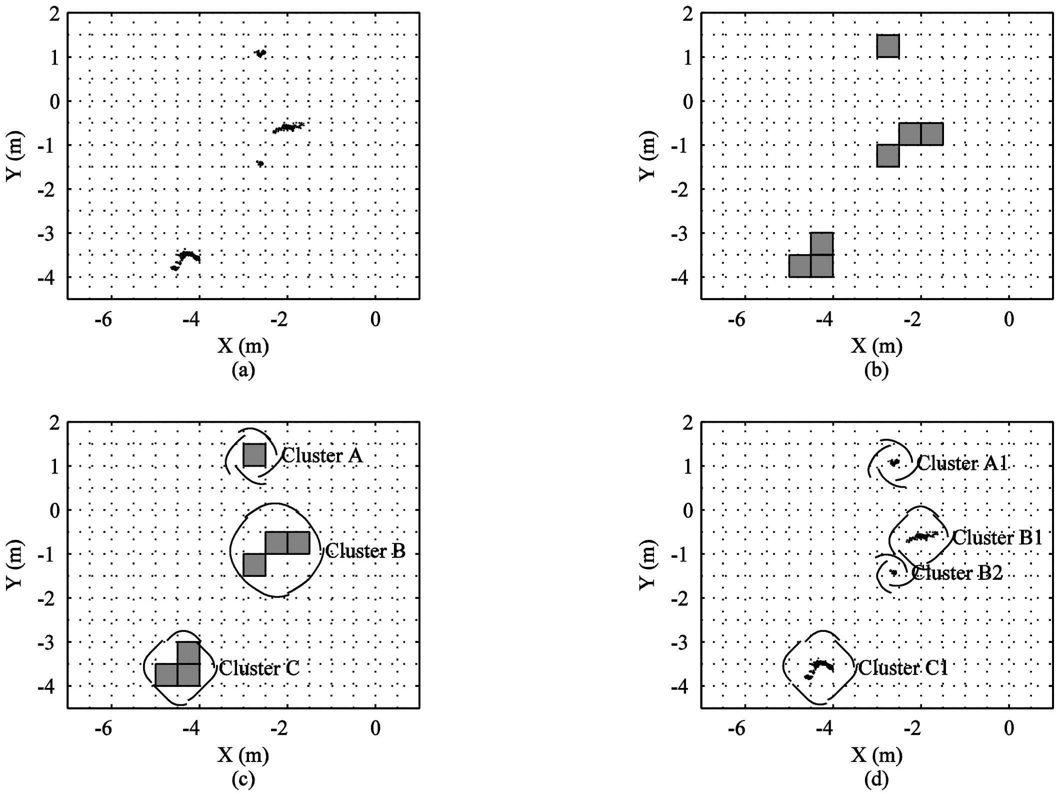


Fig. 7. Two-stage clustering example: (a) initial data, (b) occupied grid cells, (c) clustered grid cells, (d) final clusters.

2.6.3 Stem Modeling with Cylinders and Cones

To model tree stems given a cluster of candidate main stem data points near breast height, best-fit cylinders and cones are computed. The fits are found by solving a traditional least-squares problem, where n is the number of points in the cluster and f_i is the distance from the i^{th} point to the surface of the cylinder/cone (i.e. residual):

$$\underset{\mathbf{x}\in\Re^3}{\arg\min}\frac{1}{2}\sum_{i=1}^{n} f_i^2(\mathbf{x}) \cdot \tag{10}$$

For this work, Matlab’s lsqnonlin function is used for optimization.

Here, the cylinder is represented by five parameters: the angles to rotate the cylinder axis (about the roll and pitch axes) to vertical, è=[α, β]; the x and y coordinates of the intersection of the cylinder with the X-Y plane, $\mathbf{p_t}$=[x_t, y_t]; and the cylinder radius, r. Using these parameters, it is straightforward to compute the transformation between inertial coordinates and cylinder-centered coordinates, where the z axis is aligned with the cylinder axis. In this formulation, the sum of squared residuals, s, can be calculated as

$$s = \sum_{i=1}^{n} \left(\sqrt{x_i^2 + y_i^2} - r \right)^2 , \tag{11}$$

where (x_i, y_i, z_i) is the i^{th} data point in cylinder-centered coordinates and r is the radius of the cylinder being fit.

To obtain least-squares cylinder results that are more realistic for trees, it is useful to constrain the optimization parameter search space. For example, the cylinder axis should be near vertical, and the radius should be constrained to prevent tree models with excessively large radii. For this work, the upper bound of the radius was set to 0.75 m. This upper bound was chosen based on the largest trunk that was measured for this work, which had a radius of 0.531 m.

The cylinder axis is more difficult to constrain due to the transformation of the cylinder axis' direction cosines, $\mathbf{a}$=[a_x, a_y, a_z], into the optimization parameters, è=[α, β]. To speed the optimization, α and β are constrained, rather than directly constraining a_z. From experimental results, it has been determined that a constraint of $|a_z|>0.9$ yields good modeling results. A direction cosine of 0.9 corresponds to an angle of approximately 26^{o} from the vertical. Trees do not grow perfectly straight, and this constraint is meant to prevent gross errors where the axis is very far from vertical. Thus, since a_z=cos(α)cos(β), for a_z to be constrained to be greater than 0.9 means that α and β will each be constrained to the ranges [-arccos(0.9), arccos(0.9)], or [-0.45, 0.45] radians.

A similar process is used to perform least-squares optimization and constraints for cones. We would expect a frustum of a cone to be a better model for the breast-height slice of a tree stem due to the natural taper of a tree stem. Here, each cone is specified by eight degrees of freedom. These include three degrees for the direction cosines of the cone axis, $\mathbf{a}=[a_x, a_y, a_z]$, three degrees for a point on the cone axis, $\mathbf{p_{init}}=[x_{init}, y_{init}, \mathrm{z}_{\mathrm{init}}]$, one degree for the cone's radius, *r*, at point $\mathbf{p_{init}}$, and one degree for the cone's vertex angle, φ (half of the included vertex).

The cone axis is constrained in the same way that the cylinder axis is constrained. The upper bound of the cone angle is set to 0.1 radians (≈6°) and the lower bound is zero. These bounds were selected based on empirical observations. This is significantly larger than the taper expected at breast height for average trees (Li and Weiskittel, 2010).

2.6.4 Rejection Criteria for Tree Models

The final step in the tree modeling algorithm is to determine the quality of the least-squares fits and reject models that are unrealistic, or poorly fit the data. Four heuristically-inspired constraints to characterize poor fits are described below. If any of these criteria are met for a given tree, it will be rejected as a poor fit. The following explanations specifically refer to cylinder fits, but the same criteria are also used for cone fits. For criteria that are based on radius, the radius of the cone at a height of 130 cm above the ground is used.

Rejection Criterion 1: Distance from center of fit to centroid of data

The first rejection criterion discriminates based on the ratio of the cylinder radius, *R*, to the distance, *r*, from the center of the cylinder to the centroid of the data (in the X-Y plane). Fig. 8 and Fig. 9 show examples of overhead views to illustrate the reasoning for this criterion. In each plot, the main stem data is gray and cylinder cross-sections (at breast height) are in black. In addition, the centroid of the data is marked with an asterisk and the center of the cylinder cross-section is marked with a "+."

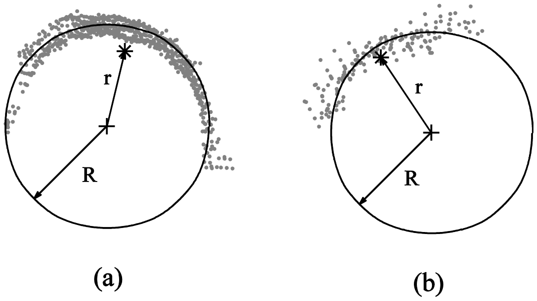


Fig. 8. Example overhead views for good cylinder/cone fits.

The plots in Fig. 8 exemplify the variability in resolution and shape of data among different trees. In Fig. 8(a), almost the entire visible surface (from the static position of the lidar) of the main stem has lidar returns, which gives the data a semicircular shape. In Fig. 8(b), only the front surface of the stem has lidar returns, which gives the data a "flat" look. It should be noted that the plots in Fig. 8 are not from the same scene and are not scaled equally. However, given the input data, both of these cylinder fits look reasonable. Also note that the distance from the center of the cylinder to the centroid of the data is slightly less than the radius in both of these cases. The ratio of the radius to the distance from the center to centroid is approximately 1.46 for the left example and 1.21 for the right.

Fig. 9 shows an example where the axis of the least-squares cylinder intersects the data. Clearly, this is not a reasonable fit, because the data does not lie on the surface of the cylinder. In this case, the center of the cylinder is very close to the centroid of the data and the ratio of the radius to the distance from the center to centroid is approximately 22.5. For this work, cylinder/cone fits are rejected if their radius to distance from center to centroid ratio is greater than 2, a value selected based on empirical data.

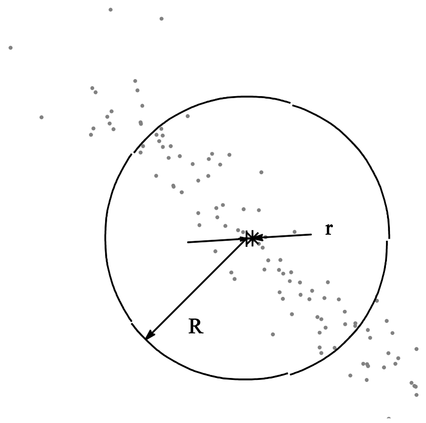


Fig. 9. Bad ratio of radius to distance from cylinder center to distance centroid.

Rejection Criterion 2: Maximum data spacing within a cluster

The next rejection criterion uses the ratio of the estimated cylinder diameter, *D*, to the maximum spacing (in the X-Y plane), *d*, of main stem data within that cluster. This is essentially a method to reject tree models with diameters that are too large, without using a naïve cutoff. A predefined cutoff is not viable because the trees in these experiments vary widely in size. Instead, the cutoff should be scaled to the size of the input data (e.g. maximum spacing of the data).

It is easy to see that for good fits, such as those in Fig. 8, the cylinder diameter is approximately equal to the maximum horizontal span of the data. Fig. 10 shows an example where the diameter of the cylinder is too large, considering the input data. In this case, the ratio of the diameter to the maximum horizontal span is approximately 2.95.

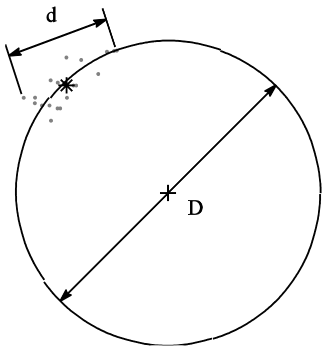


Fig. 10. Bad ratio of diameter to maximum data spacing.

While the fit in Fig. 10 looks reasonable for the given data, it grossly extrapolates the data, which has been shown to yield poor results in practice. Thus, this is another viable option for rejecting potentially erroneous least-squares fits. This problem of overestimating radius is fairly common, particularly for sparse data which lacks the expected curvature to fit a cylinder. For this work, fits are rejected if their diameter to maximum data spacing ratio is greater than 2.

Rejection Criterion 3: Intersection between tree models

There are cases where two estimated cylinders overlap each other, or an estimated cylinder is completely contained within another. This problem can be caused by a least-squares fit of excessively large radius. It can also be caused by improper clustering, where data from two nearby trees is grouped into the same cluster, or data from one tree is grouped into two clusters. Fig. 11 shows a representative situation with two overlapping cylinders.

To detect this problem we calculate pairwise distances between the centers of each cylinder, then subtract the radii of both cylinders from that distance. If this difference is negative, the two cylinders must have intersecting boundaries, or one is contained inside the other. When this happens, the cylinder with the larger radius will be rejected. In the case of an excessively large radius, or two nearby trees being grouped into the same cluster, rejecting the larger cylinder is the favorable solution because it is clear that this cylinder is breaking the constraint. In the case when data from one tree is grouped into two clusters, it is less important which fit gets rejected if the other criteria in this section are being met.

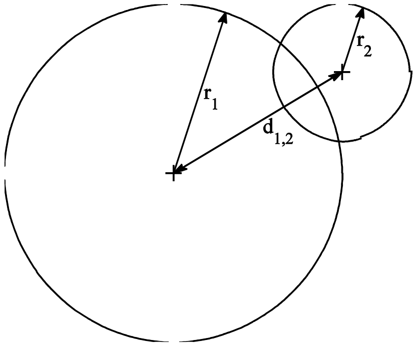


Fig. 11. Intersecting tree models.

Rejection Criterion 4: Data on the far side of trees

In some cases, a cylinder is fit such that the cylinder is positioned on the “wrong” side of the data (i.e. a surface that is physically unobservable from the sensor location). In practice this would be equivalent to a lidar sensor scanning the back of a tree (with respect to the position of the lidar), which is impossible. This problem usually arises with sparse data exhibiting indistinct curvature.

Detection of this condition can be performed by comparing the distance from the lidar to center of the cylinder with the distance from the lidar to the centroid of the stem data. If the data is on the “back” of the cylinder, the distance to the centroid will be greater than the distance to the center of the cylinder. This difference must be greater than 0.25 cylinder radii to be rejected.

This buffer is used as a margin of safety to reduce the possibility of erroneously rejecting good fits.

An example of this is shown in Fig. 12. The lidar sensor is marked by the black dot at the origin, and the cylinder in the upper-left corner marked with an arrow is clearly positioned on the "wrong" side of the data with respect to the sensor. The distance between the centroid and cylinder center is approximately 0.76 of the cylinder radius, so this cylinder would be rejected as a poor fit.

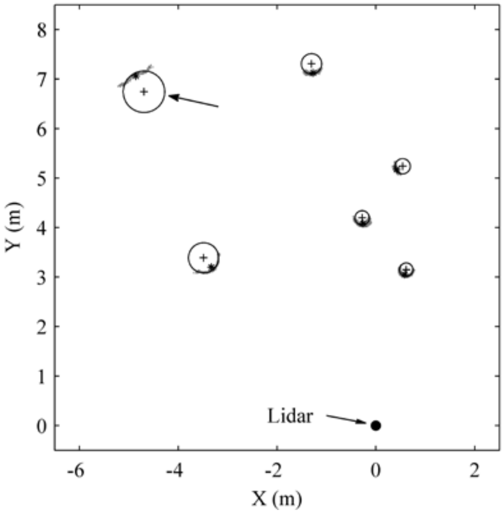


Fig. 12. Cylinder position on wrong side of data.

3. Results and Discussion

3.1. Ground Plane Estimation Results

3.1.1 Stage 1

The two-stage approach for identifying points on the ground plane was applied to the five experimental data sets described previously in this paper. On average, the local height-based filter eliminated 98.77% of the data points from consideration. Thus, this filter drastically reduced the number of points to be analyzed. To assess the effectiveness of this method, it is useful to analyze what types of data points were present in each grid cell. For example, columns that are farther away from the lidar may not have any ground points because the ground is occluded by trees and shrubs. However, if a column contains both ground and non-ground points, this filter should select a ground point. These results are summarized in Table 2. Here accuracy is defined as the fraction of columns containing both ground and non-ground points in which the true label of the lowest point was ground. True labels were determined using the approach presented in Section 2.4.

As expected, there are many columns in each scene that contain only non-ground points, unavoidably leading to error. However, in columns containing both ground and non-ground points, this filter stage selected ground points an average of 98.65% of the time.

Table 2

Stage 1 Height Filter: Column Analysis

Scene	**True Label of Lowest Point**	**Columns with Only Ground Data**	**Columns with Only Non-Ground Data**	**Columns with Both Ground and Non-Ground**	**Accuracy in Columns with Both**
baseline	Ground	542	0	135	100%
	Non-Ground	0	261	0	
sparse	Ground	509	0	622	99.35%
	Non-Ground	0	1174	4	
moderate	Ground	251	0	193	97.41%
	Non-Ground	0	1176	5	
dense1	Ground	815	0	384	97.66%
	Non-Ground	0	313	9	
dense2	Ground	694	0	521	98.66%
	Non-Ground	0	352	7	
Totals	Ground	2811	0	1855	98.65%
	Non-Ground	0	3276	25	

3.1.2 Stage 2

The second stage in the ground plane identification approach uses an SVM classifier to assign each of the minimum *z* points as either ground or non-ground. The results are presented in Table 3. For each data set, the SVM was trained using all other data sets. For example, the *baseline* scene was trained on the *sparse*, *moderate*, *dense1* and *dense2* scenes; the *sparse* scene was trained on the *baseline*, *moderate*, *dense1* and *dense2* scenes; etc. In Table 3, the quality of the classifier is assessed using the positive predictive value (PPV), which is calculated as the number of ground points correctly classified divided by the total number of points classified as ground.

Table 3

Stage 2 Classifier

		True Label		
Scene	**SVM Classification Result**	**Ground**	**Non-Ground**	**Positive Predictive Value (PPV)**
baseline	Ground	666	10	98.52%
	Non-Ground	11	248	
sparse	Ground	1032	163	86.36%
	Non-Ground	99	1011	
moderate	Ground	441	443	49.89%
	Non-Ground	3	738	
dense1	Ground	1131	71	94.09%
	Non-Ground	68	251	
dense2	Ground	1077	86	92.61%
	Non-Ground	138	272	

The results suggest that the classifier can accurately identify points lying on the ground surface in 4 of the 5 test scenes. The average positive predictive value over the five data sets is 84.29%. Fig. 13 shows the lidar data classified as ground by the SVM. This includes true positives (i.e. ground points correctly classified as ground), false positives (i.e. non-ground points incorrectly classified as ground), and false negatives (i.e. ground points incorrectly classified as non-ground).

These results show that the proposed algorithm works well for sloped terrain. Both the *dense1* and *dense2* scenes have considerable slope, with maximum ground slopes of 44.7% and 78.9%, respectively, and total elevation changes of 4.2 m and 3.1 m, respectively. The point clouds in Fig. 13(d) and (e) show that there were more false negatives at higher elevations. However, most of these points were near the edge of the sensor range where data was sparse and thus were penalized by feature f_1, which counts the number of occupied columns in the neighborhood of column (i,j). Despite the fact that the false negative rate was higher for higher elevations, enough ground points were correctly classified at these higher elevations to allow a rough estimate of ground height to be calculated.

The least accurate results were for the *moderate* scene, where the positive predictive value was 49.89%. Examining the numerical results (Tables 2 and 3) and the associated point cloud (Fig. 13(c)), it can be seen that there is significant ground cover immediately in front of the lidar scanner. Since there were no ground points in front of the sensor, bush/shrub data points were selected by the minimum z filter. This can be verified by analyzing Table 2. Compared to other scenes, there were relatively few columns in the *moderate* scene that contained only ground points, and there were many columns that contained only non-ground points. However, there were only five instances where non-ground points were chosen in columns that contained both ground and non-ground points. Since the ground cover was so low (in z height) in this scene, bush/shrub data was frequently identified as ground by the SVM. It should be noted that this was the densest scene acquired in the summer, so with respect to understory foliage it was more challenging than the *dense1* and *dense2* scenes.

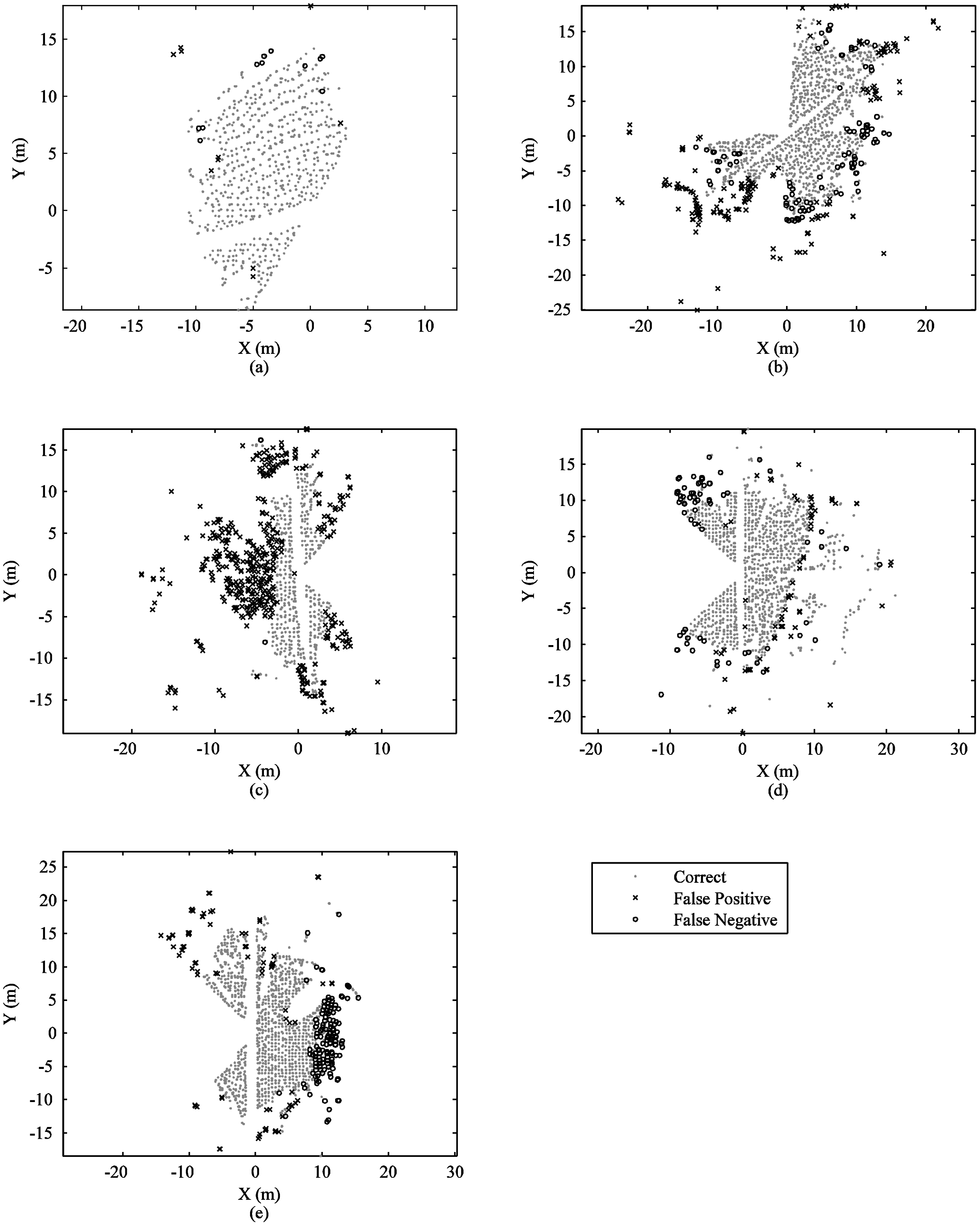


Fig. 13. Ground plane point cloud for (a) *baseline* (b) *sparse*, (c) *moderate*, (d) *dense1* and (e) *dense2* scenes.

If a lower rate of false positives is desired, as would be the case with the *moderate* scene, the threshold of the SVM classifier can be tuned accordingly. Fig. 14 shows the receiver operating characteristic (ROC) curves for the five data sets. In this figure, true positive rate and false positive rate are plotted for a range of classifier thresholds. If the threshold is high, few points will be labeled as ground, and both true positive and false positive rates will be low. If the threshold is low, many points will be labeled as ground, and both true positive and false positive rates will be high. Random assignment of points to classes would yield a line from (0%, 0%) to (100%, 100%).

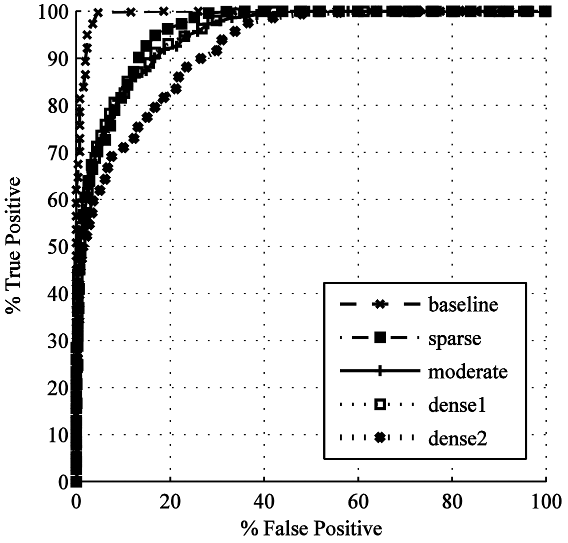


Fig. 14. ROC curves for *baseline, sparse, moderate, dense1* and *dense2* scenes.

In Fig. 14, it can be observed that low false positive rates can be achieved while maintaining relatively high true positive rates. For example, for the *baseline* scene, a 70% true positive rate can be achieved with less than a 0.8% false positive rate. The same 70% true positive rate can be achieved with less than 5% false positives for the *sparse, moderate* and *dense1* scenes. A 70% true positive rate can be achieved with less than 10% false positives for the *dense2* scene.

The SVM threshold can be tuned depending on the user's accuracy requirements. Tuning can reduce the number of false positives while retaining as many true positives as possible, at the expense of not classifying all available data.

3.2. Stem Estimation Results

3.2.1 Linkage Clustering Results

60 cm slices of candidate stem data are clustered using the divide and conquer linkage-based clustering algorithm described earlier. For this work, a cutoff distance of l=0.5 m was used for clustering subset data.

Table 4 summarizes the clustering results by comparing the number of potential clusters (identified as a connected cluster of data points by a human expert) within the convex hull of ground data to the number of clusters that the algorithm computed. Note that the number of potential clusters do not correspond exactly to the number of trees from Table 1, because some potential clusters contain purely non-trunk data. This table also quantifies the two types of errors that can occur during clustering. One error occurs when two distinctly different groups of points (e.g. points from two different trees) are grouped into the same cluster. This typically occurs when two clusters lie very close to each other. The other type of error occurs when points from one group are split into two different clusters. This typically occurs when there is a discontinuity in the data due to occlusion. The error rate is computed as the number of errors divided by the number of potential clusters.

Table 4

Linkage Clustering Performance

			Errors		
Scene	**Potential Clusters**	**Clusters Found By Algorithm**	**Two Clusters Grouped into One**	**One Cluster Grouped as Two**	**Error Rate**
baseline	4	4	0	0	0%
sparse	6	6	0	0	0%
moderate	18	19	0	1	5.55%
dense1	63	57	6	0	9.52%
dense2	40	40	0	0	0%
Totals	131	126	6	1	5.34%

Since this algorithm experienced errors caused by data from multiple trees being too closely spaced (i.e. two clusters grouped into one cluster) and data from a single tree being too widely spaced (i.e. one cluster grouped into two clusters), it can be concluded there is no static cutoff distance that can ameliorate all of these clustering errors. Empirical performance has shown 0.5 m to be a reasonable choice for the cutoff distance. Future research into an adaptive clustering threshold could yield techniques to provide better clustering performance.

3.2.2 Tree Modeling Results

Each cluster is fit to a cylinder and cone using constrained least-squares optimization. The resulting least-squares cylinder and cone fits are plotted for the *dense1* and *dense2* datasets in Fig. 15 and Fig. 16, respectively. The plots contain the final results after the rejection criteria are applied to the fits. The location of the lidar in each scene is shown with a star.

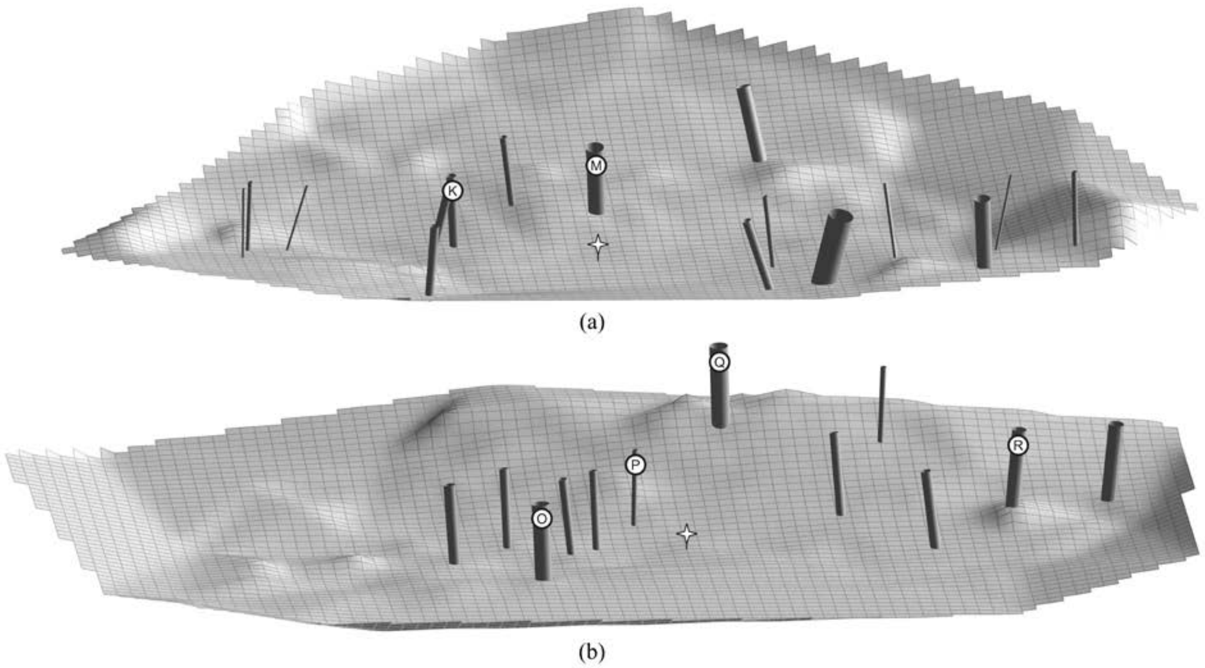


Fig. 15. Cylinder fit tree models and ground model for (a) *dense1* and (b) *dense2* scenes.

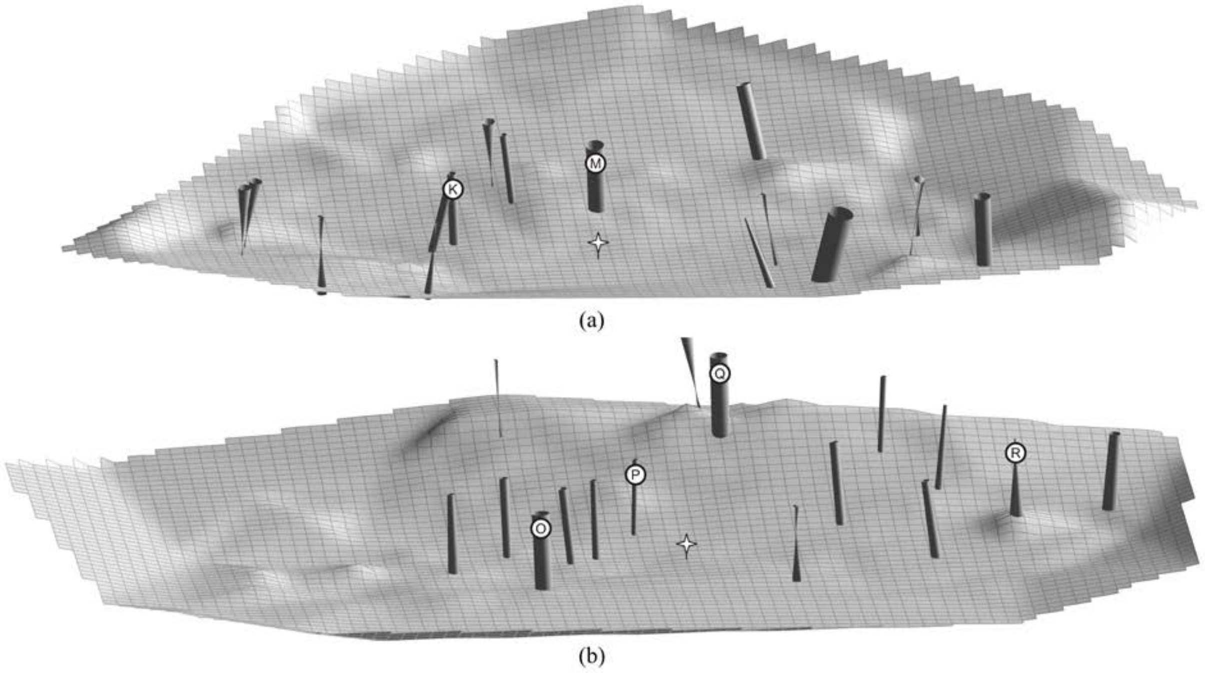


Fig. 16. Cone fit tree model and ground model for (a) *dense1* and (b) *dense2* scenes.

Table 5 summarizes the number of trees that were successfully modeled using cylinders and cones. The number of trees include of all stems with data in the initial breast-height slice of data. Any fit that was not eliminated by the rejection criteria as an unrealistic fit is considered successfully modeled.

Table 5

Number of Trees Successfully Modeled Using Cylinders and Cones

Scene	**Number of Trees**	**Number of Trees Successfully Modeled Using Cylinders**	**Percent of Trees Successfully Modeled Using Cylinders**	**Number of Trees Successfully Modeled Using Cones**	**Percent of Trees Successfully Modeled Using Cones**
baseline	4	3	75.0%	4	100.0%
sparse	5	2	40.0%	3	60.0%
moderate	16	6	37.5%	8	50.0%
dense1	48	16	33.3%	16	33.3%
dense2	40	12	30.0%	16	40.0%
Totals	113	39	34.5%	47	41.6%

3.2.3 Slice Width Analysis

The size of the slice of candidate stem data has a significant effect on the accuracy of the final tree modeling results. The average model accuracy and number of modeled trees are plotted as a function of the slice width in Fig. 17 and Fig. 18, respectively.

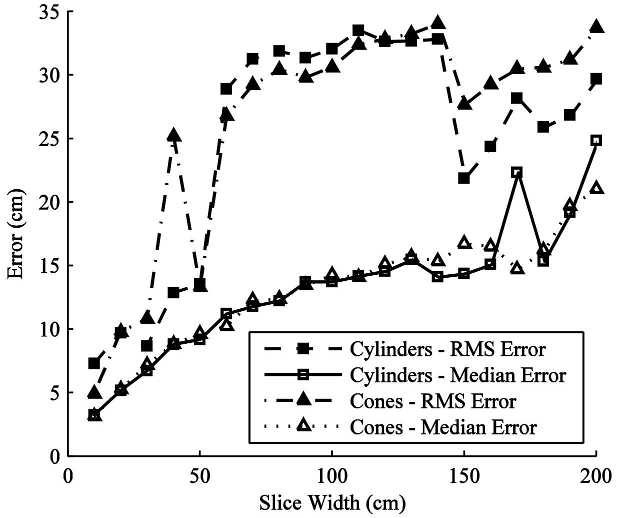


Fig. 17. Diameter error vs. slice width.

Fig. 17 plots the root mean square (RMS) and median error of the final modeling results as a function of the slice width. The error refers to the absolute difference for each tree between the dbh estimated from the fit model and the actual (hand-measured) dbh. Here we can see a clear trend of increasing error with increasing slice width. While a few large errors introduce significant variation in the plot of RMS error, the median exhibits a nearly monotonic upward trend. Thus, to achieve the minimum error, a small slice width is desirable.

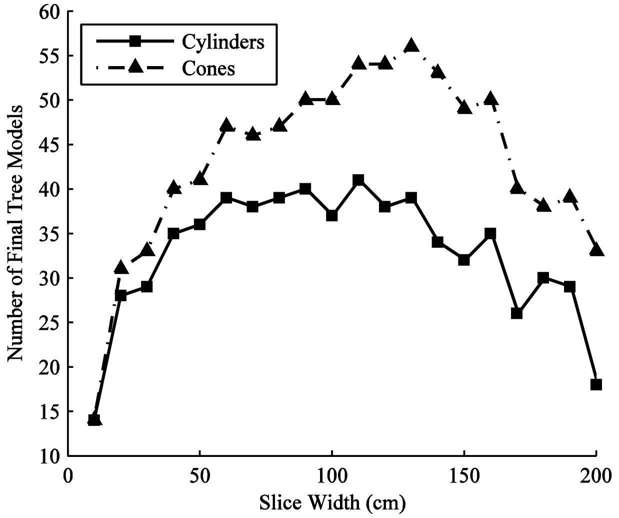


Fig. 18. Number of modeled trees vs. slice width.

Fig. 18 plots the final number of trees that were modeled (after the rejection criteria were applied) as a function of slice width, out of a total of 113 actual trees. These plots have a parabolic trend. For cylinders, slice widths between 60 cm and 130 cm model similar numbers of trees. For cones, widths between 60 cm and 160 cm model similar numbers of trees. For smaller widths, less data is available to compute the least-squares fits. This causes some of these fits to be inaccurate, and thus more likely to be rejected according to the criteria in Section 2.6.4. Meanwhile, larger slice widths include more data, which is likely to include data points from canopy and understory. This extraneous data causes erroneous least-squares fits, which are eliminated by the rejection criteria, so all of the final tree models represent actual trees. Based on the number of trees fit, the optimal slice width is between 60 cm and 130 cm for cylinders, and between 60 cm and 160 cm for cones.

Fig. 17 shows that errors are similar when comparing cylinder and cone estimates. However, Fig. 18 shows that cones successfully model more trees for every slice width. For this reason, cones have superior overall performance compared to cylinders. This is to be expected, since tree main stems exhibit a natural taper, and thus cones represent a fundamentally more accurate representation of tree stem geometry.

3.2.4 Range Cutoff Analysis

As plotted in Fig. 17, the RMS errors for main stem diameter estimates for cylinders and cones were approximately 28.9 cm and 26.8 cm, respectively, using a 60 cm slice width. Meanwhile, the median error was 11.2 cm for cylinders and 10.2 cm for cones. The individual errors are plotted as a function of distance from the lidar in Fig. 19. This plot contains the absolute value of the errors as well as a linear best-fit to show the trend of the data.

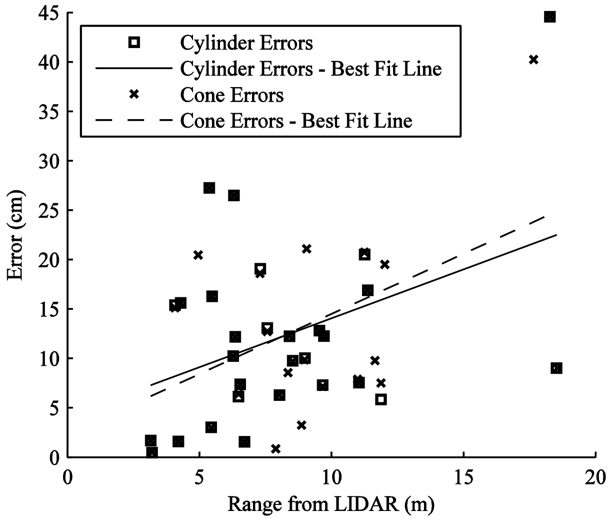


Fig. 19. Stem estimation error as a function of range from the lidar.

Fig. 19 shows a trend that stem estimation errors increase at greater ranges from the sensor. This is to be expected because stems that are closer to the sensor tend to have a higher point density and yield a more accurate fit. Meanwhile, data that are far from the sensor frequently suffer from sparseness and/or occlusion.

A few of the points in Fig. 19 show relatively large errors (>15 cm) for ranges less than 7 m. These errors are due to either low branching of the main stem or partial occlusions by other trees and vegetation. Low branching can cause large variations in the stem diameter over a small change in height, potentially causing discrepancies between the measured and estimated diameter. When a tree is partially occluded, the algorithm relies on sensor data from only a small segment of the tree’s perimeter to estimate its diameter, making the algorithm sensitive to small variations in the shape of the tree’s cross section.

It should be noted that a single outlier—an error of 135 cm at a range of 14 m—was removed from this plot so that the other points could be more easily observed. This outlier was the result of a tree directly in line with the pitch axis of the nodding device that was used to collect the lidar data. This resulted in sharp discontinuities and data only being collected on the lower portion of the trunk, which led to a least-squares fit with a large radius. In practice, the models of trees located in line with the pitch axis should be treated with caution until they can be sensed with the lidar in a different orientation.

Based on the evidence in Fig. 19, it is beneficial to use only part of the full range of lidar data to obtain more consistent and accurate stem estimates. Here we consider a cutoff set at a range of 13 m from the lidar sensor (i.e. no data beyond 13 m is considered). Fig. 20 and Fig. 21 show the accuracy and number of trees modeled trees with this 13 m cutoff, out of a maximum of 113 actual trees.

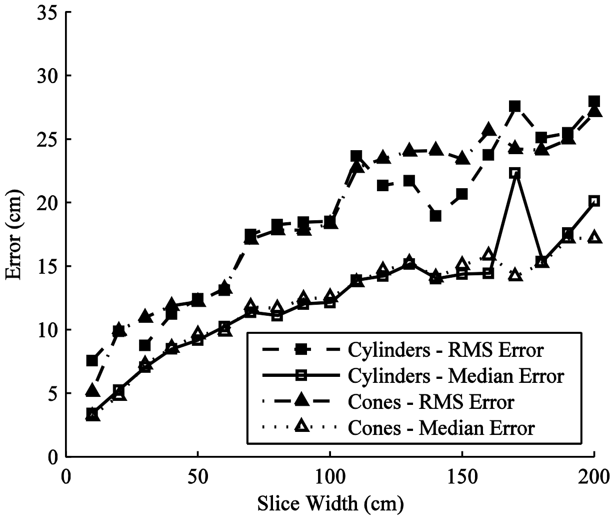


Fig. 20. 13 m Cutoff: Error vs. slice width.

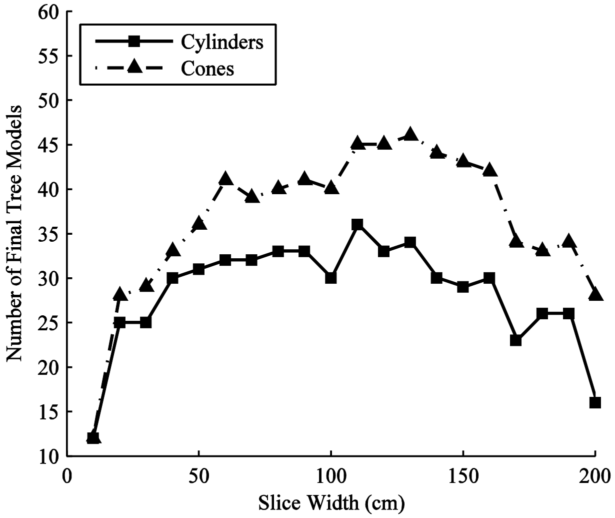


Fig. 21. 13 m Cutoff: Number of modeled trees vs. slice width.

Fig. 20 shows that the RMS errors with the 13 m cutoff are significantly lower than without the cutoff (from Fig. 17). The RMS and median errors are compared in Table 6. The 13 m cutoff has very little effect on the number of final trees modeled, according to Fig. 21. This is expected because most of the data greater than 13 m from the sensor is too sparse to result in an accurate least-squares fit.

Table 6

RMS and Median Diameter Errors Using Cylinders and Cones

Range of Data Utilized	RMS Error Using Cylinders	Median Error Using Cylinders	RMS Error Using Cones	Median Error Using Cones
No cutoff	28.9 cm	11.2 cm	26.8 cm	10.2 cm
13 m cutoff	13.1 cm	10.2 cm	13.2 cm	9.8 cm

4. Conclusions

This paper has presented a unified, novel approach for classification and modeling of forested terrain using remotely sensed lidar data. Particularly, this work focuses on ground plane and stem estimation. Ground plane estimation is performed using a local height-based filter to eliminate a large percentage of non-ground points. A novel SVM-based supervised classification approach operating on eight geometrically defined features identifies which of the remaining points belong to the ground. The stem modeling technique first selects a slice of lidar data centered at a height of 130 cm off the ground. This data is then clustered using an efficient linkage-based clustering algorithm. Trees are modeled by fitting the clustered data to cylinders and cones, and constraints are introduced to reject unrealistic fits. This approach was tested using lidar data collected during experiments from five distinct forested environments. Results demonstrate that this approach can accurately and reliably model both the ground and tree stems in conditions with sparse ground cover, such as open forested environments in summer or denser environments in winter. Errors in ground height estimation may occur when low groundcover is too dense to be penetrated by lidar.

The algorithm presented here does have limitations, including the inability to model tree stems leaning more than 26°, due to restrictions in the assumed fits. While the dbh estimates are not as accurate as some previously reported (e.g. Henning & Radtke (2006)), this is to be expected when using a lidar scanner that weighs only 370 g and is easily mounted on a man-portable robot. Additionally, the single-scan tree detection rate exceeds that reported elsewhere using a significantly larger and heavier unit (Thies & Spiecker, 2004). Additionally, due to the reliance on lidar data from a single viewpoint, this algorithm can only model trees in the sensor's line of sight—trees which are hidden at breast height will not be modeled.

Future work will involve extending this method to a moving vehicle, where it can be used for many applications. Both the ground plane and obstacle (stem) locations must be known for safe path planning of a UGV. Stem locations and diameters can also be used for UGV localization. The ability to collect this information from a moving platform would also be valuable in forestry applications, such as mapping and forest characterization. It is expected that robust performance from a moving vehicle will depend on accurate estimation of the relative sensor pose between one scan line and the next.

Acknowledgements

This work was supported by the US Army Research Office (contract #W912HZ-08-C-0060), and the US Army Topographic Engineering Center (contract #W911NF-07-1-0540).

References

Aschoff, T., Thies, M., Spiecker, H. (2004, July). Describing forest stands using terrestrial laser-scanning, in: International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 35(B5). Presented at the XXth ISPRS Congress, Geo-Imagery Bridging Continents, Commission V, Istanbul, Turkey, 237-241.

Carle, P., Furgale, P., Barfoot, T. (2010). Long-Range Rover Localization by Matching LIDAR Scans to Orbital Elevation Maps. Journal of Field Robotics, 27(3), 344-370.

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

Forsman, P., & Halme, A. (2005). 3-D mapping of natural environments with trees by means of mobile perception. IEEE Transactions on Robotics, 21(3), 482-490. doi:10.1109/TRO.2004.838003

Henning, J., Radtke, P. (2006). Detailed stem measurements of standing trees from ground-based scanning lidar. Forest Science, 52, 67-80.

Jain, A. K., Dubes, R. C. (1988). *Algorithms for Clustering Data.* Prentice-Hall Advanced Reference Series. Prentice-Hall, Inc., Upper Saddle River, NJ, 58-89.

Lalonde, J.-F., Vandapel, N., Huber, D.F., Hebert, M. (2006). Natural terrain classification using three-dimensional ladar data for ground robot mobility. Journal of Field Robotics 23(10), 839-861.

Lefsky, M., Harding, D., Cohen, W., Parker, G., Shugart, H. (1999). Surface lidar remote sensing of basal area and biomass in deciduous forests of eastern Maryland, USA. Remote Sensing of Environment, 67, 83-98.

Lefsky, M., Hudak, A., Cohen, W., Acker, S. (2005). Geographic variability in lidar predictions of forest stand structure in the Pacific Northwest. Remote Sensing of Environment, 95, 532-548.

Li, R., Weiskittel, A.R. (2010). Comparison of model forms for estimating stem taper and volume in the primary conifer species of the North American Acadian Region. Annals of Forest Science 67, 302.

Lucas, R., Lee, A., Williams, M. (2006). Enhanced simulation of radar backscatter from forests using lidar and optical data. IEEE Transactions on Geoscience and Remote Sensing, 44, 2736-2754.

Maas, H.-G., Bienert, A., Scheller, S., Keane, E. (2008). Automatic forest inventory parameter determination from terrestrial laser scanner data. International Journal of Remote Sensing 29, 1579-1593.

Manduchi, R., Castano, A., Talukder, A., & Matthies, L. (2005). Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation. Autonomous Robots, 18(1), 81-102. doi:10.1023/B:AURO.0000047286.62481.1d

Pradalier, C., Sekhavat, S. (2003). Simultaneous Localization and Mapping using the Geometric Projection Filter and Correspondence Graph Matching. Advanced Robotics, 17, 675-690.

Quick Terrain Modeler (2009, June). Applied Imagery. Computer software version 6.1.2. Available: http://appliedimagery.com/

Raber, G., Jensen, J., Schill, S., Schuckman, K. (2002). Creation of digital terrain models using an adaptive lidar vegetation point removal process. Photogrammetric Engineering & Remote Sensing, 68(12), 1307-1315.

Schölkopf, B. (2000). Statistical Learning and Kernel Methods (Technical Report No. MSR-TR-2000-23). Microsoft Research.

Sharma, M., Parton, J. (2009). Modeling Stand Density Effects on Taper for Jack Pine and Black Spruce Plantations Using Dimensional Analysis. Forest Science 55, 268-282.

Thies, M., Spiecker, H. (2004). Evaluation and future prospects of terrestrial laser-scanning for standardized forest inventories. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 36 (Part 8/W2). 192-197.

Thies, M., Pfeifer, N., Winterhalder, D., Gorte, B.G.H. (2004). Three-dimensional reconstruction of stems for assessment of taper, sweep and lean based on laser scanning of standing trees. Scandinavian Journal of Forest Research 19, 571 - 581.

Vandapel, N., Donamukkala, R., Hebert, M. (2006). Experimental Results in Using Aerial LADAR Data for Mobile Robot Navigation, in: Field and Service Robotics, Springer Tracts in Advanced Robotics. Springer Berlin / Heidelberg, pp. 103-112.

Wellington, C., Courville, A., Stentz, A. (2005, June). Interacting Markov Random Fields for Simultaneous Terrain Modeling and Obstacle Detection. Proceedings of Robotics: Science and Systems, Cambridge, MA.

Wellington, C., Stentz, A. (2003, July). Learning Predictions of the Load-Bearing Surface for Autonomous Rough-Terrain Navigation in Vegetation. Field and Service Robotics Conference, Lake Yamanaka, Japan.