11-25-2020

Traversability analysis in unstructured forested terrains for off-road autonomy using LIDAR data

Morteza Foroutan

Traversability analysis in unstructured forested terrains for off-road autonomy using LIDAR data

By

Morteza Foroutan

Approved by:

Wenmeng (Meg) Tian (Committee Chair)
Kari Babski-Reeves (Co-advisor)
Brian Smith
Mohammad Marufuzzaman
Linkan Bian (Graduate Coordinator)
Jason M. Keith (Dean, Bagley College of Engineering)

A Dissertation
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Industrial and Systems Engineering
in the Department of Industrial and Systems Engineering

Mississippi State, Mississippi

November 2020

Name: Morteza Foroutan

Date of Degree: November 25, 2020

Institution: Mississippi State University

Major Field: Industrial and Systems Engineering

Committee Chair: Wenmeng (Meg) Tian

Title of Study: Traversability analysis in unstructured forested terrains for off-road autonomy using LIDAR data

Pages in Study: 121

Candidate for Degree of Doctor of Philosophy

Scene perception and traversability analysis are real challenges for autonomous driving systems. In the context of off-road autonomy, there are additional challenges due to the unstructured environments and the existence of various vegetation types. It is necessary for the Autonomous Ground Vehicles (AGVs) to be able to identify obstacles and load-bearing surfaces in the terrain to ensure a safe navigation (McDaniel et al. 2012). The presence of vegetation in off-road autonomy applications presents unique challenges for scene understanding: 1) understory vegetation makes it difficult to detect obstacles or to identify load-bearing surfaces; and 2) trees are usually regarded as obstacles even though only trunks of the trees pose collision risk in navigation.

The overarching goal of this dissertation was to study traversability analysis in unstructured forested terrains for off-road autonomy using LIDAR data. More specifically, to address the aforementioned challenges, this dissertation studied the impacts of the understory vegetation density on the solid obstacle detection performance of the off-road autonomous systems. By leveraging a physics-based autonomous driving simulator, a classification-based machine learning framework was proposed for obstacle detection based on point cloud data

captured by LIDAR. Features were extracted based on a cumulative approach meaning that information related to each feature was updated at each timeframe when new data was collected by LIDAR. It was concluded that the increase in the density of understory vegetation adversely affected the classification performance in correctly detecting solid obstacles. Additionally, a regression-based framework was proposed for estimating the understory vegetation density for safe path planning purposes according to which the traversabilty risk level was regarded as a function of estimated density. Thus, the denser the predicted density of an area, the higher the risk of collision if the AGV traversed through that area. Finally, for the trees in the terrain, the dissertation investigated statistical features that can be used in machine learning algorithms to differentiate trees from solid obstacles in the context of forested off-road scenes. Using the proposed extracted features, the classification algorithm was able to generate high precision results for differentiating trees from solid obstacles. Such differentiation can result in more optimized path planning in off-road applications.

DEDICATION

I dedicate my dissertation work to my family. A special feeling of gratitude to my loving wife, Fatemeh whose words of encouragement and her belief in my abilities helped me endure the difficult times we both have been through during my PhD studies. My parents Mohammad Hossein and Rahimeh who gave me moral lessons on discipline from an earlier age and inspired me to pursue my education and earn a doctorate degree.

ACKNOWLEDGEMENTS

I wish to thank my committee members who were patient with me and generous with their expertise and time. A special thanks to Dr. Wenmeng Tian, the chair of my committee for her countless hours of reading, reflecting, and guiding me throughout the entire research process. She exemplifies a true advisor who goes above and beyond to ensure her students doing quality work. Also, I would like to extend my sincere thanks to Dr. Kari Babski-Reeves for her precious support from the start of my PhD studies. She provided continuous guidance and helped me get through the difficult life events I experienced during my PhD studies. Thank you, Dr. Mohammad Marufuzzaman, and Dr. Brian Smith for agreeing to serve on my committee.

I would like to acknowledge and thank Dr. Christopher Goodin from the Center for Advanced Vehicular Systems at Mississippi State University whose tremendous technical support made it possible for me to make progress throughout the research process. Finally, I would like to thank Dr. Mohammad Mehrmohammadi from Wayne State University who assisted me, encouraged me and provided mental support that made the completion of this research an enjoyable experience.

TABLE OF CONTENTS

APPENDIX

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Vehicles capable of driving autonomously have several applications in different environments including “indoor robotics to unmanned commercial and military vehicles and interplanetary exploration” (Nefian and Bradski 2006). Waymo, Tesla, General Motors and others have demonstrated vehicles which can reliably interpret human-placed infrastructure, such as lane markings and signs, and navigate accordingly (General Motors 2018; TESLA 2020; Waymo 2018). Several benefits are associated with the use of autonomous vehicles, such as elimination of human-error related road accidents, optimized energy consumption and comfort (Choi et al. 2012). Reduced pollution levels and congestion are other benefits of driverless vehicles (Silva et al. 2017). In addition, there are advantages of using autonomous vehicles in military applications, such as operating in potentially dangerous environments (Epshtein and Faint 2019).

However, to benefit from the aforementioned advantages of self-driving vehicles, some challenges need to be reliably addressed. One challenge is to identify the location of the vehicle at every point in time (Choi et al. 2012) to be able to identify the path to destination. Another challenge is the ability to detect vehicle’s surrounding environment to avoid crashes (Choi et al. 2012). When considering autonomous vehicle operation in urban or rural areas, another challenge is the ability to detect traffic signs, lanes, bumps, etc. (Choi et al. 2012). Finally, an educated way of optimizing the route based on the three previously addressed challenges is required (Silva et al. 2017). Based on these challenges, Silva et al. (2017) suggest that an autonomous vehicle architecture should encompass three technological subsystems as follows:

1- **Sensing and Perception**. This system uses sensors to collect data to understand the surrounding environment. The information generated by this system will be used for basic movement decisions such as obstacle detection, moving direction, etc.

2- **Localization and Mapping**. This system uses satellite data or other methods to capture the vehicle's global location at any point in time. "Therefore, localization is performed by mapping the surrounding environment in 3D in various settings, and comparing against historical data" (Silva et al. 2017).

3- **Driving Policy System**. This system uses algorithms to make proper decisions related to autonomous driving in different situations, such as yielding or overtaking another vehicle.

Although Waymo, Tesla, General Motors and others have demonstrated relatively reliable performance in scene understanding in highly structured environments, such as highways, a lack of comparable prior work exists in the context of unstructured areas. This research is specifically focused on the applications of autonomous ground vehicles (AGV) in unstructured forested areas. The US department of Defense, through its various agencies, has been the major sponsor of research in this field (Manduchi et al. 2005). To autonomously drive in a safe manner, AGVs must be able to identify the load-bearing surface of the terrain (i.e. the ground) and obstacles in the environment (McDaniel et al. 2012). However, estimating the load-bearing surface based on the LIDAR data can be challenging due to the existence of vegetation such as grass, shrubs, trees. Also, not all the surfaces identified as ground are safe for driving. For example, there might be tree branches or other solid obstacles that would occupy part of the space above the identified 'drivable ground'. Additionally, the ability to make a distinction between the vegetation and other obstacles is not always sufficient to guarantee safe navigation (Ahtiainen, Stoyanov, and Saarinen 2017). Thus, this research is an effort in the context of an unstructured forested environment to provide a robust solution for identifying not-safe-to-drive areas where solid obstacles such as rock or tree trunks exist as opposed to drivable areas where either no obstacle or soft obstacles such as shrubs or thin tree branches exist.

1.1 Motivation and Background

Elimination of human operators as a bottleneck and reducing the operations costs are some benefits of using autonomous navigation in the forested environment (Hellström et al. 2009). The "salary of operators generally amounts to 30 to 40 percent of the hourly cost of a forest machine" (Hellström et al. 2009). In the context of farming, considering the competitive nature of today's businesses, it is obvious that reducing operational costs can provide a competitive edge to those who own a technology enabling them to reduce headcount and eliminate these associated costs. In the context of military applications, another important benefit to the use of autonomous vehicles can be added as they can eliminate or reduce the risk of endangering individuals' lives who would have been physically present in the vehicle. Existence of such technologies also provides a better opportunity for more frequent and more aggressive exploratory missions. Those opportunities, however, come with their own challenges. One challenge is the identification of a drivable surface (McDaniel et al. 2012). The existence of traffic signs, traffic lights and lines can facilitate the identification of the safe driving path for the autonomous driving in the urban areas. However, off-road autonomous vehicles may be required to operate based on algorithms to enable the vehicle to safely navigate the terrain in the absence of such cues. One intricacy of identifying obstacles in an unstructured forested terrain (as an example of an off-road application of AGVs) is that the vegetation such as shrubs, grass, main tree stems, and tree crowns can be detected as obstacles -adding unnecessary limitations to the path identification and planning- since some of those identified obstacles could be ignored in the path planning process. For instance, tree crowns or shrubs may pose a reduced threat whereas tree trunks can severely damage an AGV in case of contact. Thus, making a distinction between ground and non-ground areas and making a distinction between a safe non-ground area as opposed to non-safe ones are problems that need to be addressed in order to achieve a reliable and safe autonomous driving capability.

This dissertation is motivated by the aforementioned challenges and the practical needs for safe off-road autonomous applications. Thus, the purpose of this dissertation is to develop and validate a robust approach that can be used in identifying safe-to-drive terrains in unstructured forested areas in the context of autonomous unmanned ground vehicle.

1.2 Research Objectives

The overarching goal of this research was to define strategies and techniques that can aid AGVs in identifying solid obstacles that might pose collision risks in unstructured forested areas. The research objectives were to:

- Assess the impact of understory vegetation density on solid obstacle detection for off-road autonomous ground vehicles.
- Conduct traversability risk analysis through the development and validation of a predictive model to estimate the understory density.
- Develop and validate of an algorithm to identify both trees and other solid obstacles in forested terrains.

1.3 Definition of Key Terminologies

Key terminologies used in this dissertation are defined below:

- **LIDAR**: is an airborne laser scanning technology which stands for “Light Detection and Ranging” (Reutebuch, Andersen, and McGaughey 2005). For autonomous vehicles to map the surrounding environment, laser scanners emit high rate pulses of near-infrared laser beams. The distance (range) between the vehicle and objects reflecting the emitted pulses are then calculated considering the travel time for each pulse between the scanner and the object (Reutebuch, Andersen, and McGaughey 2005). “The sensor readings result in a set of 3D points, also called Point Cloud (PCL), and corresponding reflectance values representing the strength of the received pulses” (Arnold et al. 2019).
- **Localization**: in the context of self-driving vehicles, it is the capability of an autonomous vehicle to identify its position with regard to its environment (Pendleton et al. 2017).
- **Perception**: is a capability of an autonomous system to gather data, process it, and generate relevant knowledge from the surrounding environment (Pendleton et al. 2017). Environmental perception refers to “developing a contextual understanding of environment, such as where obstacles are located” (Pendleton et al. 2017).
- **Autonomous Vehicle**: is a type of vehicle that has “the capability to perceive the surrounding environment and navigate itself without human intervention” (Jo et al. 2014). In this study, the terms ‘autonomous vehicles’, ‘unmanned vehicles’, ‘driverless vehicles’, ‘self-driving vehicles’, and ‘robotic vehicles’ can be used interchangeably.

1.4 Dissertation Outline

This dissertation has seven chapters in total. CHAPTER II provides a comprehensive literature review for the state-of-the-art work relevant to this dissertation. CHAPTER III describes the approach used to conduct this research, introduces the simulation platform and describes the datasets used throughout the study. CHAPTER IV introduces a methodology to assess the impact of understory vegetation density on solid obstacle detection for off-road AGVs. This chapter was developed as a research paper which is accepted by *ASME Letters in Dynamic Systems and Control*[1]. CHAPTER V introduces a data-driven framework to assess traversability risk through the estimation of understory density, which was developed as a research paper submitted to the *Journal of Autonomous Vehicles and Systems*[2]. CHAPTER VI proposes a data-driven methodology for identification of trees in an obstacle-rich forested environment. CHAPTER VII discusses the contribution of this dissertation and CHAPTER VIII highlights future opportunities.

[1] Link to the journal: www.journaltool.asme.org/home/JournalDescriptions.cfm?JournalID=35&Journal=ALDSC

[2] Link to the journal: www.journaltool.asme.org/home/JournalDescriptions.cfm?JournalID=37&Journal=JAVS

CHAPTER II

LITERATURE REVIEW

In this chapter, a review of state-of-the-art is provided. More specifically, prior work on modeling surrounding environment, assessing understory vegetation density, and autonomous driving simulation tools will be reviewed, and the research gaps in the state-of-the-art work are summarized.

2.1 Modeling Surrounding Environment for Off-Road Autonomy

Since an autonomous vehicle operation is dependent on its capability of sensing its surrounding environment using different sensors such as LIDAR, cameras, ultrasound and radar (Silva et al. 2017), one area of focus for several researchers has been developing methods and algorithms to reliably model the surrounding environment of an autonomous vehicle using one type of sensor or a combination of different sensors.

Prior research work related to autonomous driving covers a wide variety of topics ranging from perception and mapping towards planning, control, and cooperative behavior (Stiller et al. 2015). This study, however, focuses on the perception and mapping aspect of autonomous driving. More specifically, the study is concentrated on the use of LIDARs in modeling the surrounding environment to identify obstacles. Particularly, this study considers the off-road applications of AGV in the unstructured forested areas. Thus, the review of literate concentrates on the prior work that can provide insight and direction to help the advancement of this research. The relevant work can be categorized into the following three topics.

2.1.1 Terrain Map Generation

In order to generate a reliable drivable map, a representative spatial model of the surrounding environment is needed (Ahtiainen, Stoyanov, and Saarinen 2017). Thus, a lot of work in the context of autonomous driving has been focused on the special modeling such as in Bares et al. (1989); Franz et al. (2005); De Luca et al. (2008); Garrido, Malfaz, and Blanco (2013); Hornung et al. (2013); Stoyanov et al. (2013). Occupancy gridding technique is one of the most prevailing spatial modeling techniques that was originally proposed by (Moravec and Elfes 1985). Even though the occupancy grids were developed as a 2D method, they are scalable to 3D applications (Ahtiainen, Stoyanov, and Saarinen 2017). However, when dealing with large scale data, the memory requirements of such method for 3D application makes it practically impossible to use (Ahtiainen, Stoyanov, and Saarinen 2017).

OctoMap is a widely used method for implementing 3D occupancy maps which is “based on octrees and uses probabilistic occupancy estimation” (Hornung et al. 2013). Due to its multi-resolution support, this method offers an effective way to keep free and unknown portions of the map which in turn maintains memory requirements at minimum (Hornung et al. 2013).

Another mapping approach that has been used since the early years of robotics applications is elevation maps which is a 2.5D gridding method (Ahtiainen, Stoyanov, and Saarinen 2017). One downside of this method compared to OctoMap is its reduced dimensionally from 3 to 2.5 which comes from its inability to model more than one surface per cell, thus, it fails to properly model overhanging structures (e.g., bridges, tunnels, tree branches) (Ahtiainen, Stoyanov, and Saarinen 2017). Furthermore, elevation maps are not able to reliably estimate the height of areas for which little or no data is available (Ahtiainen, Stoyanov, and Saarinen 2017).

To address the problem related to missing height-data in elevation maps, Lang, Plagemann, and Burgard (2007) proposed a model that takes into account the change in data densities of terrain models. They developed the model using an adaptive nonstationary kernel regression model in Gaussian process (Lang, Plagemann, and Burgard 2007). The main idea is to represent the height value of each point as a function of its 2D space coordinates, and to subsequently approximate the value using a set of Gaussian distributions (Ahtiainen, Stoyanov, and Saarinen 2017).

Another spatial modeling method which is particularly popular among the computer graphists is called polygonal meshes (Ahtiainen, Stoyanov, and Saarinen 2017). A polygonal mesh is "a graph of interconnected vertices in which each polygon represents a facet in the mesh" (Ahtiainen, Stoyanov, and Saarinen 2017). Since generating reliable reconstruction results of the environment from the noisy cloud data requires specific treatment of data in terms of filtering and dealing with uncertainty, Wiemann et al. (2010) proposed a method that automatically develops trainable meshes from noisy point cloud data. One shortcoming of their method, though, is that the triangle mesh is developed offline and as a post-processing step, thus, is not kept on the go (Ahtiainen, Stoyanov, and Saarinen 2017).

Another special model method which was originally developed by Biber and Straßer (2003) is called Normal Distribution Transform (NDT). Similar to occupancy Grid mapping, NDT works in a 2D context, however, while offering comparable accuracy results, NDT is able to use significantly larger cell sizes (Stoyanov et al. 2013). The idea behind NDT is to model identified points (range data) as a set of Gaussian probability distributions, in other words, each identified point is being represented by a Gaussian distribution that estimates the probability of its presence in a particular physical location (Ahtiainen, Stoyanov, and Saarinen 2017).

An extension to NDT is NDT Occupancy Map (NDT-OM) which “is a 3D spatial model based on a regular grid that concurrently estimates both the occupancy and the shape distribution in each cell” (Ahtiainen, Stoyanov, and Saarinen 2017). NDT-OM calculates the probabilities and updates the models of each cell on a recursive basis which in turn provides an efficient way of environment mapping in a dynamic environment (Saarinen et al. 2013). NDT-OM requires significantly lower resolution compared to other 3D spatial modeling algorithms while keeping the accuracy competitive (Ahtiainen, Stoyanov, and Saarinen 2017). Ahtiainen, Stoyanov, and Saarinen (2017) extended the previous NDT-OM work done by Stoyanov et al. (2013) with two added features: permeability and intensity to be able to improve the existing traversability analysis and vegetation detection models.

2.1.2 Scene Classification

Another area related to perception and mapping is scene classification for traversability analysis. Several methods have been proposed to address this topic such as those in Vandapel et al. (2004); Howard et al. (2006); Bajracharya et al. (2009); Martin, Murphy, and Corke (2013). Some methods are extensions of previous work while some address different concerns related to traversability such as detecting ground and non-ground environment as in (Nefian and Bradski 2006), detecting negative obstacles such as the method proposed by Heckman et al. (2007), identifying tree stems such as the work by (McDaniel et al. 2012).

Vandapel et al. (2004) presented a terrain classification method by segmenting 3-dimensional point cloud data in vegetated environment. In order to make distinction between surfaces, curves, and points of the environment, they used local point distribution statistics that generates saliency features. They used Gaussian Mixture Model (GMM) for the training set to capture the saliencies distribution. Then, they applied a Bayesian classifier to the new on-line data to perform the vegetated terrain classification.

One improvement opportunity with the approach Vandapel et al. (2004) presented is the processing time. The other improvement is reducing the rate of classification error. The latter was achieved by "dealing with border effects and isolated range measurements" (Vandapel et al. 2004).

Manduchi et al. (2005) proposed a terrain classification method that distinguishes between solid obstacles (such as rocks or tree trunks) and grass. Using LIDAR data, their algorithm "analyzes the slant of surface patches in front of the vehicle, and identifies patches that are steep enough to represent a hurdle for the vehicle" (Manduchi et al. 2005). Their method addresses some of the shortcomings of previously developed terrain classification algorithms as most of the prior work was focused on urban or indoor environments in which the assumption of flat ground surface would work well (unlike the off-road application) (Manduchi et al. 2005). They used a combination of two systems to provide complementary functionalities for safe driving in the context of off-road navigation: 1- a color stereo camera was used to identify isolated obstacles. 2- a single axis lidar was installed bellow the vehicle to detect partially hidden obstacles behind tall grass that may not be captured by the stereo camera.

Both the Elevation maps methods that were formerly used by Lacaze, Murphy, and DelGiorno (2002) or voxel-based environment representation methods as in (Hebert et al. 2002) require change in input data structure leading to some transformation requirements which in turn requires extra computational power as well as storage space. The method Manduchi et al. (2005) proposed, however, works directly with the 3D point cloud data. To identify obstacles, their proposed method measures slope and height of "visible surface patches" (Manduchi et al. 2005).

They used two simple logics for identifying drivable paths: 1- if the slope of the surface patch is limited, it is assumed that the patch is part of the ground. 2- if the slope is steep but has a short height, it is assumed that the patch belongs to a small obstacle that can be neglected by the vehicle.

Another advantage of the method proposed by Manduchi et al. (2005) is that there is no need to use the vehicle orientation data. Also, their algorithm detects obstacles from the range data on a single-frame basis. Thus, it does not require integration of all frames, which in return adds to efficiency of their algorithm.

One drawback of the method proposed by Manduchi et al. (2005) is that isolated points sticking out from the ground may not be identified as obstacles as they do not meet the compatibility requirement introduced in their algorithm. Another critique to their proposed work is that they did not use the intensity data which could improve their model performance by incorporating information regarding different types of obstacle that might be modeled using the intensity data.

Nefian and Bradski (2006) presented methods for detection of drivable corridors for off-road autonomous navigating that grew out of the model used during the DARPA Grand Challenge Race. DARPA Grand Challenge Race is a challenge administered by the Defense Advanced Research Projects to encourage innovations related to AGVs (Buehler, Iagnemma, and Singh 2007). The authors of the article were part of the Stanford team that won the DARPA challenge in 2005 (Nefian and Bradski 2006). Nefian and Bradski (2006) used a combination of sensors including Global Positioning System (GPS), LASER, and cameras to determine the drivable corridors. They used a Hierarchical Bayesian Network (HBN) to model images by clustering them into different categories such as sky, road, and side of the roads (Nefian and Bradski 2006).

According to Nefian and Bradski (2006), embedded hidden Markov model (EHMM) is "one of the most successful statistical models used to describe a specific category of images with similar properties". However, the authors found the Bayesian Network a better choice when segmenting a larger class of images as it can "successfully account for natural dependencies between neighboring pixels in both image dimensions" (Nefian and Bradski 2006). While the accuracy of their HBN model was significantly better than an EHMM with shadow nodes, the latter showed higher runtime speed (Nefian and Bradski 2006).

One shortcoming of the model proposed by Nefian and Bradski (2006) is that it was developed and tested based on pictures captured in the Nevada dessert. Thus, same concepts cannot be applied on the forested terrains. Furthermore, they proposed a set of geometrical and smoothness constraints that were specifically tuned for the images used in the study (Nefian and Bradski 2006). Thus, it might not offer more universal solution.

Wellington, Courville, and Stentz (2006) proposed a probabilistic terrain model that works based on some main assumptions. They used two Markov Random Fields to "encode the assumptions that ground heights smoothly vary and terrain classes tend to cluster" (Wellington, Courville, and Stentz 2006). Their model also uses a latent variable to encode the assumption that "vegetation of a single type has a similar height" (Wellington, Courville, and Stentz 2006). Their assumptions might be reasonable in agricultural applications. As an example, farms are normally flat or have smooth slopes. Also, planting seeds occurs almost at the same time across a field. Thus, similar heights of vegetation are expected; However, the same assumption might fail in unstructured off-road terrains such as natural forests.

Choi et al. (2012) proposed environmental detection algorithm for rural and off-road environments. They used vision-based method to detect lane, pedestrian, and speed bumps. Additionally, they used LIDAR-based method to detect obstacles. Due to challenges concerning lack of enough GPS data points in off-road applications, Choi et al. (2012) developed an algorithm to produce a local map of obstacles. They used a height-difference algorithm along with a classification algorithm to estimate the risk of each point being an obstacle and to develop a risk map of obstacles.

A notable shortcoming of the method proposed by Choi et al. (2012) is that they developed their algorithms for static environments. The assumption of static environments is not a realistic assumption. While such an assumption simplifies the problem, it significantly limits the applications of their work.

Using 3D LIDAR data, Santamaria-Navarro et al. (2015) developed an off-line terrain classification method which applies GP to model the environment with automatically labeled dataset. To classify different areas, they used Principal Component Analysis (PCA) to compute slope and roughness (Santamaria-Navarro et al. 2015). Also, they proposed two classification approaches: the first approach only uses positive teaching samples in the GP regression collected from the vehicle footprint while the second approach requires teaching samples from both negative and positive classes (Santamaria-Navarro et al. 2015).

According to Ahtiainen, Stoyanov, and Saarinen (2017), the method proposed by Santamaria-Navarro et al. (2015) increases the classification accuracy compared to prior methods. However, it comes at the cost of being more computationally expensive. Besides, they only use two features (slope and roughness) to perform the classification which may not be always sufficient (Ahtiainen, Stoyanov, and Saarinen 2017).

Magnusson (2009) and Stoyanov et al. (2010) used NDT framework to develop their terrain traversabilty analysis models. They proposed a Constant Threshold Classification (CTC) algorithm. According to their proposed algorithm, some predefined thresholds are used to compare against the calculated roughness and inclination within each cell distribution (Magnusson 2009; Stoyanov et al. 2010).

One of the shortcomings of the CTC algorithm is its poor performance in unstructured environments. In order to address that issue, Ahtiainen, Stoyanov, and Saarinen (2017) extended the CTC work by adding two more features: intensity distribution and permeability. While the new model improved the classification accuracy of previously developed CTC algorithm, it lacks consistent results specifically in the context of densely vegetated environment.

Using LIDAR as the only environment scanning technology can be challenging "due to obstructions and occlusions caused by vegetation" (Silva et al. 2017). Therefore, Silva et al. (2017) proposed a method to fuse LIDAR data with camera in which they address two underlying issues concerning sensor data fusion:

1- Spatial misalignment between different sensors' data

2- Resolution differences between different sensors (Silva et al. 2017).

To address the first issue, Silva et al. (2017) use a geometric model to spatially align the data coming from two different sensors (LIDAR scanner and monocular image sensor). To address the second issue, they used a Gaussian Process regression algorithm to match the resolutions of different data streams. It is achieved by deriving the spatial covariance from the data generated by camera (Silva et al. 2017).

Silva et al. (2017) claim that their proposed method of using multimodal sensor data fusion improved the detection capability performance of 'free space detection' algorithm by more than 60% compared to a single sensor-based system. However, one can argue that improving the detection algorithm of a single sensor-based system might be a cheaper and easier-to-maintain solution than adding more and different sensors to the system. Furthermore, they only used SVM algorithm for the single sensor to compare it against their proposed model. Use of other algorithms might have resulted in less significant difference in the model performance.

McDaniel et al. (2012) presented a detailed paper (about 50 pages) that addresses two subjects in the context of autonomous driving in forested environments: 1- terrain classification by identifying ground vs. non-ground areas 2- tree stem identification as opposed to shrubs, grass, etc.

With regard to the ground plane estimation, McDaniel et al. (2012) used a two-step approach. Firstly, they used a local height-based filter for each grid to only keep the lowest height point as candidate ground points. Secondly, they used an SVM-based supervised classification approach with eight geometrically defined features to classify each candidate point belonging to either of the four following categories: (1) ground, (2) bushes/shrubs, (3) tree main stems and (4) tree canopy (McDaniel et al. 2012). Overall, they used five different vegetated scenes near Boston to train their models, each time they used one of the scenes as the test set and the four remaining ones as the training sets to fit and evaluate the accuracy of the proposed model.

With regard to the tree stem detection algorithm, McDaniel et al. (2012) offered a method that first takes a horizontal slice of the LIDAR data vertically centered around a predefined height from ground called ‘breast height’. “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)” (McDaniel et al. 2012).

As for they proposed method, they chose the breast height to be 130 cm above the ground. The next step, according to McDaniel et al. (2012), is modeling tree stems by fitting data into cones and cylinders. In order to do that, a vertical range containing the breast height is needed. Use of low vertical range might result in missing a tree while use of larger range might contain "unwanted non-trunk data from above (canopy) and below (shrubs, rocks, etc.)" (McDaniel et al. 2012). To make the balance between the two competing factors and considering the specifications of their LIDAR, they used a range of 40 cm (20 cm below and above the breast height). If a column with the range of 40 cm contains LIDAR data, they considered that column as tree trunk candidate. Thus, they increased the data range for that column from ± 20 cm to ± 30 cm to better fit the data in their models. Next, they used single-linkage clustering to estimate which candidate tree truck data points belong to the same tree. Then, they fit the data into cones and cylinders (representing tree trunks) by solving traditional least square problem. Finally, they used four rejection criteria to eliminate those models that poorly fit the data or do not seem realistic.

While the algorithms proposed by (McDaniel et al. 2012) provide a lot of valuable insights and significantly contribute to the previous work, they have some limitations and shortcomings as well. As an example, their algorithm cannot model tree trunks leaning more than 26°. Also, their method relies on LIDAR data from a single viewpoint which means that tree trunks hidden at the breast point cannot be modeled. Furthermore, use of the predefined value 130 cm as the breast height might not be the best way as the forest type and understory vegetation can drastically affect the optimum breast height value. Additionally, it should be noted that even if their model correctly classified a surface as ground, that does not necessarily mean that it is safe to drive on that surface, because above the same ground, there might be some thick and solid branches of tree in a height level that might cause damage to the vehicle if the vehicle drives over that surface. In other words, along with the identification of ground, any hanging or protruding obstacle should be considered for traversability analysis of any given surface.

2.1.3 Vegetation Detection

The majority of the available proposed traversability analysis methods regard detected objects as solid and static obstacles which fail to properly address objects specifically exist in the context of vegetated areas such as forested terrains (Ahtiainen, Stoyanov, and Saarinen 2017). Some work in this area was done based on the specific properties of vegetation. For example, one property of the vegetation is that while the chlorophyll (which is found in living plants) absorbs red and blue light, it reflects near-infrared light (Myneni et al. 1995). Using that property, people such as Bradley et al. (2004); Nguyen et al. (2012); Wurm et al. (2014) presented some solutions for detecting vegetations considered as chlorophyll-rich. However, using this property to detect vegetation requires technologies other than regular LIDARs such as multi-spectral cameras.

Another property of vegetation is that LIDAR rays penetrate through sparse vegetation as opposed to solid obstacles such as rocks (Ahtiainen, Stoyanov, and Saarinen 2017). Using this property, Lacaze, Murphy, and DelGiorno (2002) introduced they vegetation detection method which counts the number of laser beams that were reflected as opposed to those that penetrated through a given cell (count of hits and misses) and thus calculated the density based on those two values.

One shortcoming of the method presented by Lacaze, Murphy, and DelGiorno (2002) is that it does not address those obstacles that are partially occluded by the vegetation. Hidden obstacles can pose real collision risks. Thus, it is important that the traversability analysis models consider and address scenarios in which obstacles could get partially covered by the vegetation.

Macedo, Manduchi, and Matthies (2001) proposed a method that uses statistical analysis of the data generated by 2D LIDAR to determine if an obstacle is non-traversable such as rock or traversable such as a patch of grass. Their method uses some predefined thresholds to determine if the variance and skewness of range distribution from LIDAR data belong to a specific classification category. Their method showed good results even in the case of partially occluded obstacles.

Using a 2D LIDAR is a drawback of their method as it limits their classification capability by the lack of the third dimension of data. As an example, an obstacle can be categorized as rock but still be safely traversable due to its low height which can be detected by adding the third dimension (height) to the interpretations.

Wellington and Stentz (2004) also used 'hits and misses' method using a voxel-based approach according to which the pass-through rays of LIDAR and rays that hit the voxel would be recorded. They used an online adaptive approach to adjust the real height of ground as opposed to LIDAR readings. Furthermore, they used saliency features, some simple statistics on height values and maximum laser remission values to perform classification. The focus of their study was on the identification of the load-bearing surfaces. Thus, same shortcoming as mentioned for Lacaze, Murphy, and DelGiorno (2002) also applies here.

Similar to the approach that Lalonde et al. (2006) proposed, Ahtiainen, Stoyanov, and Saarinen (2017) developed their model using the intensity values. However, instead of using maximum intensity values as in Lalonde et al. (2006), they used intensity distributions that "better captures the true underlying intensities of cells" (Ahtiainen, Stoyanov, and Saarinen

2017). Also, the intensity distribution provides “the intensity variance as an extra feature to use in the classification” (Ahtiainen, Stoyanov, and Saarinen 2017). Besides, their model calculates slope and roughness whereas in Lalonde et al. (2006) the saliency is calculated.

Ahtiainen, Stoyanov, and Saarinen (2017) conclude that detecting obstacles behind vegetation can be detected by LIDARs as long as part of the obstacle is still visible. They identified that if the obstacle is covered by more than three layers of vegetation, the detection would fail as the obstacle would not be visible anymore. Their conclusion, however, was based on very specific experiments and limited to a few trials which might not be used as general rule of thumb.

2.2 Assessing Understory Vegetation Density

There are relatively few studies relevant to applications of LIDAR in assessing understory vegetation density. Some of these studies have used parametric modeling techniques to develop predictive models. For example, Stepwise Regression was used to characterize the relationship between aboveground biomass (AGBM) and forest structural characteristics by leveraging data collected through a large-footprint LIDAR instruments, i.e., a Laser Vegetation Imaging Sensor (LVIS), to test the ability of LIDAR(Drake et al. 2002). In addition, Multiple Regression was used in (Hudak et al. 2006) to compare the ability of LIDAR data and multispectral satellite imagery for modeling and mapping of basal area and tree density. They concluded that the LIDAR height related variables and lidar intensity related variables were the most informative variables to predict basal area, while LIDAR canopy cover variables and LIDAR intensity are most useful for tree density prediction (Hudak et al. 2006). Another parametric study applied Ordinary Least Squares Regression using features, such as mean height, mean of the 95th percentile of heights, median height, maximum height, and standard deviation of heights for estimation of above ground biomass in a tropical forest (Clark et al. 2011b).

Some other studies applied non-parametric modeling techniques. For instance, various statistics related to height data were used to develop a Random Forest model to estimate the understory shrub density for the LIDAR-based assessment of wildlife habitat suitability (Martinuzzi et al. 2009). Moreover, K-nearest Neighbor models leveraged LIDAR height related features along with digital elevation model (DEM) variables to predict tree-level inventory data in a forested area (Falkowski et al. 2010). In another non-parametric study, a Support Vector Machines model was developed based on variables derived from LIDAR data related to both elevation channels and the intensity of returned lights (Dalponte, Bruzzone, and Gianelle 2011). In addition, Campbell et al. (2018) investigated different LIDAR metrics on their capacity to quantify vegetation understory structure in understanding of wildlife habitats, nutrient cycling, wildland fire behavior, and wildland firefighter safety applications. Specifically, they compared the performance of two metrics that are commonly used in analyzing understory vegetation, i.e. Overall Relative Point Density (ORD) and Normalized relative Point Density (NRD). They concluded that NRD is a significantly more capable metric for characterizing understory density compared to ORD (Campbell et al. 2018).

Unfortunately, most of the prior studies have been carried out in the context of canopy estimation, assessment of wildlife habitats, and understating wildland fire behavior. Those solutions cannot be directly applied in the context of off-road AGV navigation due to the following reasons: 1) The LIDAR viewpoints are above the canopy which is different from the off-road navigation in which the vehicle drives on the ground, leading to completely different point cloud distributions; 2) AGVs need to make constant path planning decisions in the terrain based on data constantly collected from varying orientation and viewpoint. This requires a robust data integration method capable of constantly updating scene understanding results. Such a capability is not included in the context of canopy estimation studies; 3) Assessment of density estimation uncertainty is not deemed relevant to some topics such as wildlife habitats. Also, the lack of such assessment is observed in the prior work related to off-road autonomy such as those in Campbell et al. (2018); Clark et al. 2011a; Dalponte, Bruzzone, and Gianelle (2011); Drake et al. (2002); Falkowski et al. (2010); Hudak et al. (2006); Martinuzzi et al. (2009). Thus, a solution is needed to take into account specific conditions that an AGV deals with, such as high-frequency data collection and interpretation of data from numerous datasets from different timeframes. Also, the solution should concentrate on the understory estimation rather than canopy estimation which in turn requires extracting different features. Additionally, lack of uncertainty/risk analysis for AGVs' traversable area in the prior work regarding off-road applications needs to be properly addressed.

2.3 Simulation Environments

Modeling and simulation (MS) of off-road navigation has been around for decades, specifically to support the development of military ground vehicles (Goodin, Carrillo, and Mcinnis 2017). That is because high-fidelity simulation systems enable "safe, inexpensive, and hardware-free prototyping and testing of control systems and sensor integration tasks" (Hudson et al. 2018). In the context of off-road autonomy, simulation of the vehicles traversing complex environments requires reliable modeling of vehicle-environment interactions (Young, Kysar, and Bos 2020). While the usefulness of MS tools in evaluating and predicting the mobility performance of the off-road ground vehicles has become evident as their use have been continued over the past decades, they have not been able to reliably predict the performance of

AGVs (Goodin, Carrillo, and Mcinnis 2017). The reason is that “the limiting factor for the performance of autonomous systems is typically not the dynamic capability or the off-road mobility of the platform, but rather the sensing used by the autonomy system and the algorithms employed to process the sensor data” (Goodin, Carrillo, and Mcinnis 2017). To address such limitation in current mobility models, a simulator that is capable of accurately capturing the impact of sensors and the environment on robotic performance is deemed to be required (Goodin, Carrillo, and Mcinnis 2017). Another drawback is that the simulation of the environment, the vehicle or their interactions may not be realistic enough, leading to high-accuracy classification results when making prediction on simulated scenes, but resulting in inaccurate predations in real-world scenarios (Chavez-Garcia et al. 2018).

Over the last 15 years, several robotics simulations tools have been developed such as Gazebo (Koenig and Howard 2004) and USARSIM (Balakirsky et al. 2006; Carpin et al. 2007). Durst et al. (2012) developed a simulation tool named Autonomous Navigation Virtual Environment Laboratory (ANVEL) that attracted special attention in military robotics uses due to its simplicity in installation, application, and its capability to produce highly elaborated digital terrains (Goodin, Carrillo, and Mcinnis 2017). To be more instrumental for developers, all these tools perform the simulation on the real-time basis which has its own drawbacks such as limiting “the fidelity, physics, and realism of the sensor, vehicle, and environment simulations” (Goodin, Carrillo, and Mcinnis 2017). While those limitations might not be of great concerns for developers, the realism and reliability of the model has major importance for those responsible for testing and evaluation (T&E) of robotic capability (Goodin, Carrillo, and Mcinnis 2017).

To address data fidelity and model realism concerns, (Goodin, Carrillo, and Mcinnis 2017) proposed a different approach called Virtual Autonomous Navigation Environment

(VANE) which is an AGV simulator that does not require simulation on a real-time basis. Rather, it applies “the most realistic physics simulations for the physical processes impacting the robot, ensuring that the simulation is both realistic and predictive (Goodin, Carrillo, and Mcinnis 2017). VANE consists of physics-based simulation models for the vehicle, sensors, and the environment. The software can be used for predicating the performance of AGVs by providing a “realistic vehicle dynamics and terrain and environment simulations” (Goodin, Carrillo, and Mcinnis 2017).

Another autonomous vehicle simulator that can be used in the context of forested terrains is developed by Mississippi State Center for Advanced Vehicular Systems, CAVS. The platform is called Mississippi State Autonomous Vehicle Simulation, or MAVS. MAVS is a “3D autonomous vehicle simulator that includes a detailed physics-based LIDAR simulation” (Goodin et al. 2019). According to Christopher Goodin who has been a key player in both VANE and MAVS projects, MAVS is superior to VANE in just about every regard, but especially with respect to simulation speed. As an example, MAVS has integrated support for vehicle simulation, whereas VANE no longer does. Also, MAVS has a Python interface, making it much easier to work with, while VANE only has a C++ interface. Additionally, accessibility to VANE is extremely limited. Only those people are allowed to use VANE that have an active contract with Department of Defense specifying the necessity of their access to the tool. When the contract ends, they must delete all copies of VANE. However, MAVS is free for academic use.

2.4 Summary of Research Gaps

This chapter reviews previous work related to Spatial Modeling and Traversability Analysis. More specifically, it reviews prior work related to different terrain map generation techniques, scene object classification, solid objects and vegetation differentiation, understory density estimation, and simulation software. Based on the review of the literature, the following gaps are identified in the literature. These are the topics that have not been thoroughly addressed, and in some cases little prior relevant work has been identified:

- **Solid obstacle vs. vegetation differentiation:** The majority of the available proposed traversability analysis methods regard detected objects as solid obstacles which fails to properly address objects specifically exist in the context of vegetated areas. Such generalization of detected objects can unnecessarily narrow down the traversable paths leading to unoptimized path planning results.
- **Understory vegetation impact on the accuracy of terrain objects classification:** The existence of understory vegetation might affect the accuracy of classification algorithms. The extent of such an impact, however, has not been widely and thoroughly assessed.
- **Understory vegetation density estimation:** solid obstacles can hide behind dense vegetation. The ability to reliably estimate the density of understory, coupled with the learnings from the impacts of understory density on the accuracy of classification models can lead to more reliable traversability models. However, no prior work was found that would propose a robust framework to quantify the vegetation density in off-road autonomy applications.
- **Traversability risk map based on classification accuracy estimation:** not only areas containing solid obstacles pose collision risk, poorly classified objects could pose collision risk as well. It is because a solid obstacle could be classified as ground or understory vegetation that in turn would lead to sever collision. The prior work in the context of terrain map generation has been mainly focused on reliably classifying different objects within the terrains. In other words, they identify one risk category (i.e. solid obstacles) while they fail

to address another risk category which is the accuracy of classification itself. Again, learnings from the impacts of understory density on the accuracy of classification models can be used in developing risk maps that consider both of the aforementioned risk categories.

CHAPTER III

RESEARCH DIRECTIONS AND EXPERIMENTAL SETUPS

This chapter introduces the simulation platform used to validate the proposed methodology. In addition, the data format generated by the simulation platform and assumptions used for the entire dissertation are discussed.

3.1 Simulation Platform

This research used a physics-based autonomous vehicle simulator developed by Mississippi State University called the Mississippi State Autonomous Vehicle Simulation (MAVS). By using MAVS, densely vegetated environments can be generated based on different ecosystems. Also, it provides the capability to generate real-time LIDAR sensing data. Additionally, MAVS can be used to test the developed off-road autonomous navigation algorithms with minimal cost. Figure 3.1 provides some images representing different environmental conditions that can be simulated by MAVS. The user has the ability to define the environmental condition parameters (e.g. the amount of rain in terms of millimeters per hour), vehicle physics (e.g. powertrain and tire specifications), terrain characteristics (e.g. trees, understory plants and surfaces with modifiable dimensions and parameters).

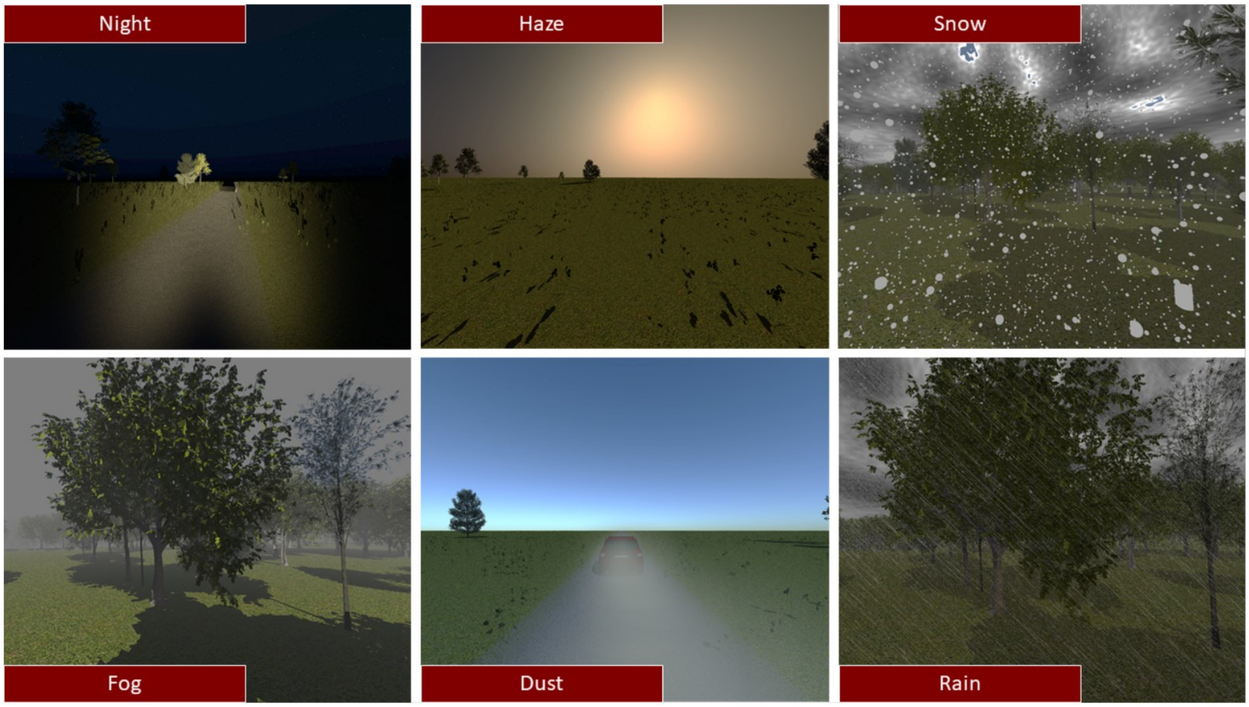


Figure 3.1 Visual examples of different environmental conditions generated by MAVS.

3.2 Dataset Description

For the purpose of this study, three different sources of data generated by the simulator were used. Those data sources are:

1- Vehicle Position: At each time frame, one data point with seven dimensions is recorded (including x, y and z coordinates and four coordinates representing the orientation of the vehicle as a quaternion). APPENDIX A shows an example of Vehicle Positions data.

2- LIDAR Readings: At each time frame, point cloud data were recorded to a text file with each row representing one point with its 3D coordinate and the reflection intensity. APPENDIX B shows an example of data file containing LIDAR readings in a certain time frame.

3- Ecosystem Objects: The coordinates and dimensions of all objects in the terrain (including plants and solid obstacles) were recorded. APPENDIX C shows an example of data file containing coordinates, length and width of bounding box of objects in a simulated ecosystem that was used in this dissertation.

3.3 Assumptions

The major assumptions of the research in this dissertation included:

1- The simulation software provided environments that are accurate enough to represent the real-world conditions.

2- No noise or other external factors such as extreme weather conditions were considered in this study to affect the accuracy of LIDAR readings.

3- Terrains were flat plains.

CHAPTER IV

ASSESSING IMPACT OF UNDERSTORY VEGETATION DENSITY ON SOLID OBSTACLE DETECTION FOR OFF-ROAD AUTONOMOUS GROUND VEHICLES

4.1 Introduction

In autonomous driving systems, advanced sensing technologies (such as LIDARs and cameras) can capture high volumes of data for real-time traversability analysis. Off-road autonomy is more challenging than other autonomous applications due to the highly unstructured environment with various types of vegetation. The understory with unknown density can create extremely challenging scenarios such as concealing potential obstacles in the terrain, leading to severe vehicle damage, significant financial loss, and even passenger injury or death. Also, negative obstacles i.e. regions that lie below the ground surface such as holes or ditches can be masked by dense vegetation and should be “treated as obstacles”(Heckman et al. 2007). This chapter investigated the impact of understory vegetation density on obstacle detection in off-road traversability analysis. By leveraging a physics-based autonomous driving simulator, a machine learning based framework is proposed for obstacle detection based on point cloud data captured by LIDAR. It was observed that the increase in the density of understory vegetation adversely affects the classification performance in correctly detecting solid obstacles. With the cumulative approach used in this study, however, sensitivity results for different density levels converged as the vehicles incorporated more time frame data into the classification algorithm. This chapter contains the content of a paper (Foroutan, Goodin, and Tian 2020) accepted by *ASME Letters in Dynamic Systems and Control* in July 2020.

4.2 Material and Methods

4.2.1 Simulation Setup

4.2.1.1 Random Terrain Construction

The randomly generated scenes are designed to be 100 meters by 100 meters. The understory vegetation used for the scenes is called couch grass and can grow up to 80cm in the terrain (Figure 4.1). The density of the understory is specified by the average number of couch grass plants in each square meter of the scene and can be controlled in the terrain construction. In addition, a straight pathway is generated in each terrain. The path starts 21 meters before entering the area covered with understory vegetation. The pathway ends 21 meters after it completes traversing through the area covered with understory vegetation. Figure 4.2 shows a 2D demonstration of a test scene developed for this study.

Figure 4.1 Couch Grass (Galium aparine)

This type of plant was used to generate different understory vegetation density for the training and testing scenes.

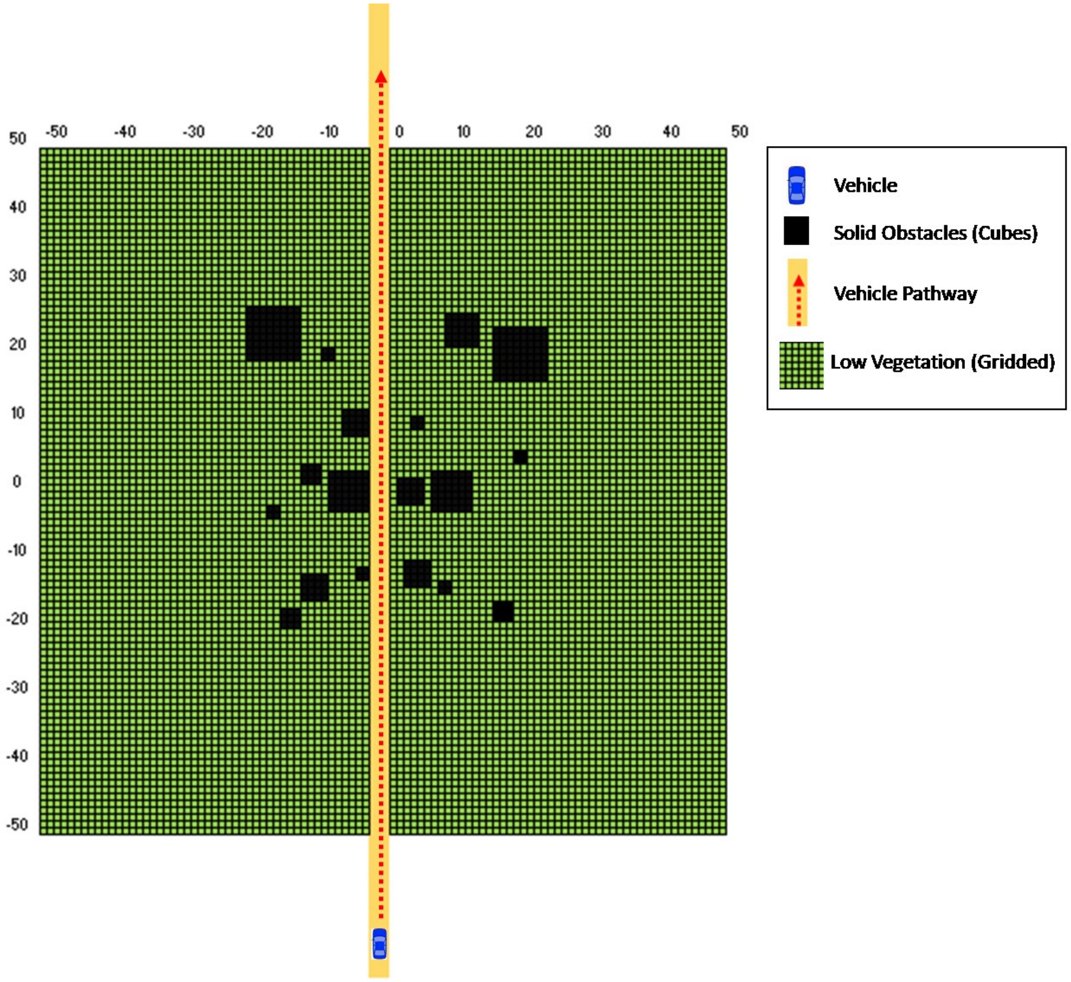


Figure 4.2 2D demonstration of designed test scene

Each cell in the gridded space represents a 50cm by 50cm square for which summary statistics are calculated. Numbers represent coordinates of the map.

To simulate solid obstacles, eighteen cubes with different dimensions were placed in random locations on the terrains. The reflecting characteristics of the cubes are set to be similar to rocks or boulders, which are significantly different from the chlorophyll-rich substance that is typically found in understory vegetation.

4.2.1.2 Sensor configuration

The LIDAR used for the simulation in this study is HDL-64E S2 developed by Velodyne Lidar[3]. The LIDAR has 64 lasers/detectors with 360° field of view. It provides user selectable frame rate that ranges from 5HZ to 15 HZ, which can facilitate high demanding perception uses as well as mobile data collection and mapping application in complicated environments. Without loss of generality, the vehicle's speed is set to a constant speed of 15 meters/sec, and the framerate is set to 10 HZ. Figure 4.3shows some visual outputs generated by MAVS for the sparse vegetated scene for one time frame.

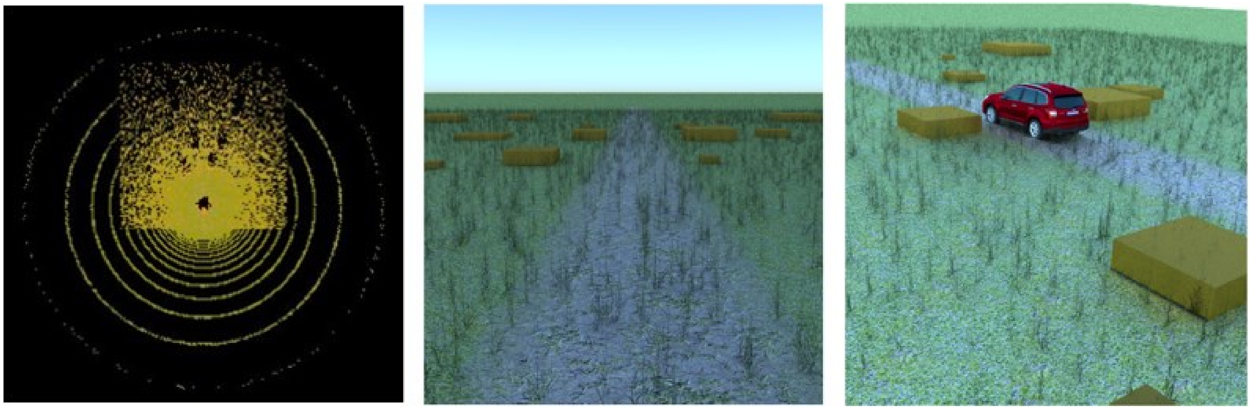


Figure 4.3 Visual Outputs generated by MAVS

The right image shows the location of the vehicle on the scene. It is following a straight path surrounded by understory vegetation and cubes with different sizes. The middle image shows the same scene from the vehicle window's view. This is specifically useful when the simulator uses manual navigation commands by the user. The left image shows the LIDAR sensor results with LIDAR in the center of the image.

[3] Link to Velodyne Lidar www.velodynelidar.com

4.2.1.3 Experimental design

Six scenes with three different density levels were developed in MAVS (Table 4.1). For each density level, one scene was used to train the generated data, and the second scene was used to test the performance of algorithm for detecting solid obstacles. Since the purpose of the study is to assess the effect of understory vegetation density, locations and sizes of cubes on the three testing terrains are identical across different density levels. Thus, the only variable is the density of the understory vegetation. The three levels of density are defined as below:

1- Density 1 (Sparse): an average of 2.5 plants per square meter

2- Density 2 (Medium): an average of 12.5 plants per square meter

3- Density 3 (Dense): an average of 25 plants per square meter

Table 4.1 Scene designs with respect to vegetation density and cube sets

	Density 1	**Density 2**	**Density 3**
Training Scene	Cube set 1	Cube set 1	Cube set 1
Testing Scene	Cube set 2	Cube set 2	Cube set 2

Figure 4.4 through 4.6 show high-resolution rendering of each vegetation density for the same time frame in the testing scene. The pictures clearly show as the density level increases, the cubes (solid obstacles) become less visible for the human eye.

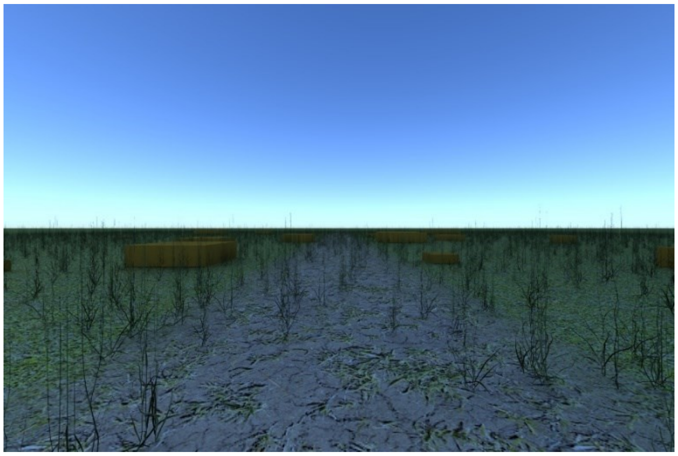


Figure 4.4 Test Scene, Density Level 1 (Sparse)

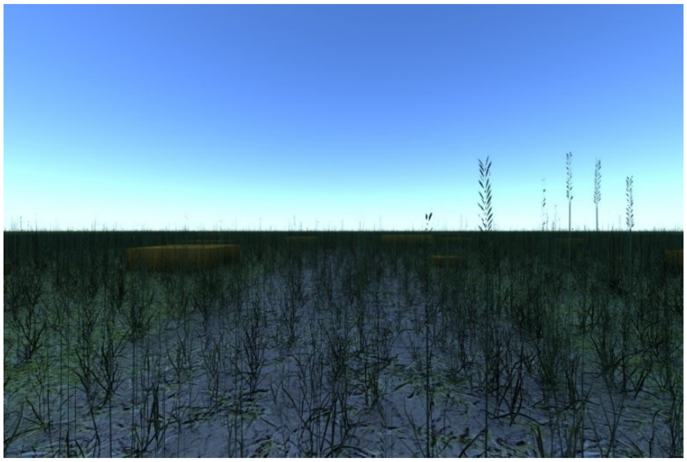


Figure 4.5 Test Scene, Density Level 2 (Medium)

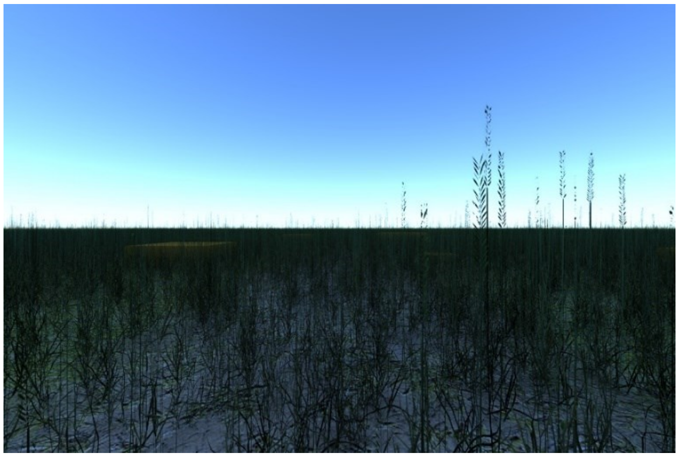


Figure 4.6 Test Scene, Density Level 3 (Dense)

4.2.2 Proposed Methodology

One challenge with using LIDAR data is that the amount of data can be excessively large in a way that the system cannot practically handle data in a real-time manner. Reducing the data size while keeping the essential information can significantly reduce the processing time, enabling the practical applications. Therefore, it is desirable to structure the unstructured point cloud data by grouping the points into regular sized cells. In this study, a cell-generating approach is proposed to reduce the data volume. Subsequently, several features are extracted from the points in each cell to preserve critical information for navigation. Two machine learning algorithms are used to unveil the systematic patterns in navigable cells and non-navigable (i.e. obstacle) cells based on the cell-wise features, and the trained classifier can be used for traversability analysis in real time.

4.2.2.1 Cell generation

The entire terrain is first segmented into square cells in the x-y plane. Given predefined cell size, s, the points fall into the *ith* row and *jth* column (denoted as *C(i,j)*) can be defined as

$$\{(x_k, y_k, z_k) | x_L + (j-1) \times s \le x_k \le x_L + j \times s \ and \ y_L + (i-1) \times s \le y_k \le y_L + i \times s\} \tag{4.1}$$

where

$$i = 1,2, \ldots, \left\lceil \frac{y_U - y_L}{s} \right\rceil \tag{4.2}$$

and

$$j = 1,2, \ldots, \left\lceil \frac{x_U - x_L}{s} \right\rceil \tag{4.3}$$

x_L and y_L are the smallest x and y values in the entire dataset while x_U and y_U are the largest values, respectively. The cell size can be determined based on the predefined navigation resolution. For this study, cell sizes of 50 cm by 50 cm was chosen. However, in real-world applications, the cell size can be determined based on the predefined control and navigation accuracy. It is worth noting that that too small cell sizes would increase the processing time while too large cell sizes would reduce the resolution for navigation and path planning.

4.2.2.2 Cell-wise feature extraction

Once the whole scene is divided into cells, the summary statistics for each cell including maximum height, minimum height, number of points, average and variance of ray intensity, and the distance of the LIDAR is calculated for each grid cell. In addition, the cell label can be calculated based on the ground truth data, which can be used as label information for each cell.

In this study, a cumulative approach is taken to update the summary statistics for each cell as the vehicle traverses through the terrain. In this sense, the information from all historical data can be kept, and the learning accuracy over the entire terrain will improve as the vehicle collects more and more data. Below are the steps taken to extract features from each cell for each time frame:

I. **Number of points related features.** Count the number of detected points *(N)* in each cell. Also calculate the cumulative average of *N* for each cell. The cumulative average of *N* for *C(i,j)*, denoted as $Mean[N(i,j)]$, can be calculated for the kth data frame as:

$$Mean[N_k(i,j)] = Sum[N_k(i,j)]/k \tag{4.4}$$

where

$$Sum[N_k(i,j)] = Sum[N_{k-1}(i,j)] + N_k(i,j) \tag{4.5}$$

II. **Ray intensity related features.** Calculate the cumulative mean and variance of ray intensity (Cum_μ_I and $Cum\_\sigma_I^2$) for each cell.

II.a. The cumulative mean of ray intensity for $C(i,j)$, denoted as $Cum\_\mu_I\,(i,j)$, can be calculated for the *k*th data frame as:

$$Cum\_\mu_{I_k}(i,j) = \frac{Cum\_Sum_{I_k}(i,j)}{Sum[N_k(i,j)]} \tag{4.6}$$

where

$$Cum\_Sum_{I_k}(i,j) = Cum\_Sum_{I_{k-1}}(i,j) + Sum_{I_k}(i,j) \tag{4.7}$$

is the cumulative summation of all intensity values pertaining to *C(i,j)*.

II.b. The cumulative variance of ray intensity for *C(i,j)*, denoted as $Cum\_\sigma_I^2(i,j)$, is calculated for the kth data frame as:

$$Cum\_\sigma_{I_k}^2(i,j) = \frac{Cum\_SS_{I_k}(i,j) - Sum[N_k(i,j)]\times Cum\_\mu_{I_k}(i,j)^2}{Sum[N_k(i,j)]} \tag{4.8}$$

where $Cum\_SS_{I_k}(i,j)$ is the cumulative sum of squares of intensity values pertaining data located in $C(i,j)$ for the k_{th} data frame. $Cum\_SS_{I_k}(i,j)$ is calculated as follow:

$$Cum\_SS_{I_k}(i,j) \; = Cum\_SS_{I_{k-1}}(i,j) \; + SS_{I_k}(i,j)$$

where $SS_{I_k}(i,j)$ is the sum of squares of intensity values pertaining to data located in $C(i,j)$ for the *k*th data frame. (4.9)

III. Height related features. Calculate the cumulative mean and variance of z values (Cum_μ_z and $Cum\_\sigma_z^2$) for each cell.

III.a. The cumulative mean of z values for $C(i,j)$, denoted as $Cum\_\mu_z(i,j)$, can be calculated for the *k*th data frame as:

$$Cum\_\mu_{z_k}(i,j) = Cum\_Sum_{z_k}(i,j)/Sum[N_k(i,j)] \tag{4.10}$$

where

$$Cum\_Sum_{z_k}(i,j) = Cum\_Sum_{z_{k-1}}(i,j) + Sum_{z_k}(i,j) \tag{4.11}$$

is the cumulative summation of all z values pertaining to $C(i,j)$ for the *k*th data frame. Also, $Sum_{z_k}(i,j)$ is the summation of all the z values pertaining to $C(i,j)$ for the *kth* data frame.

III.b. The cumulative variance of z values for *C(i,j)*, denoted as $Cum\_\sigma_z^2(i,j)$, can be calculated for the kth data frame as:

$$Cum\_\sigma_{z_k}^2(i,j) = \frac{Cum\_SS_{z_k}(i,j) - Sum[N_k(i,j)] \times Cum\_\mu_{z_k}(i,j)^2}{Sum[N_k(i,j)]} \tag{4.12}$$

where $Cum\_SS_{z_k}(i,j)$ is the cumulative sum of squares of z values pertaining to $C(i,j)$ for the *kth* data frame. $Cum\_SS_{z_k}(i,j)$ is calculated as follow:

$$Cum\_SS_{z_k}(i,j) = Cum\_SS_{z_{k-1}}(i,j) \ + SS_{z_k}(i,j) \tag{4.13}$$

where $SS_{z_k}(i,j)$ is the sum of squares of z values pertaining to $C(i,j)$ for the k_{th} data frame.

4.2.2.3 Obstacle identification based on extracted features

Two classification methods that use completely different statistical approaches were implemented. The idea was to select, fine-tune and proceed with the algorithm that provides better performance given our study data and setting. The two classification algorithms are Linear Discriminant Analysis (LDA) and Boosted Trees (BT).

To analyze the performance of the aforementioned classification methods, a number of model evaluation metrics including Accuracy, Precision, Recall, and Specificity will be used. In this study, reducing 'False Negative' cases are more important than reducing 'False Positive' cases. The reason is that False Negative means that the algorithm failed to detect a solid obstacle, which may potentially lead to a serious collision. The false positives, on the other hand, indicate that the algorithm classified a non-solid obstacle as a solid obstacle, leading to a narrowed down traversable region for navigation. Thus, in this study, while evaluating different performance measures of developed models, the focus of data analysis would be on the Recall (Sensitivity) results.

Below is a summary of main steps of the proposed obstacle identification method:

1- Locate those cells that contain solid obstacles (cubes) and label them as 1 which represents 'Obstacle'

2- Eliminate ground cells (cells that have maximum height lower than 25 cm)

3- Label the remaining grids as 0 which represents 'Low Vegetation'

4- Use the two classification methods i.e. LDA and BT to train the data (use $Mean]$, Cum_μ_I, $Cum\_\sigma_I^2$, Cum_μ_z and $Cum\_\sigma_z^2$ as predictors, use label as response).

5- Compare the performance of the classification algorithms and choose the best one (use ‘sensitivity’ as the main criterion).

6- Optimize selected classification with regards to false negatives (add penalty term to reduce false negatives).

7- Fit the tuned algorithm on the test scene and capture the performance results.

4.3 Results and Discussion

The data are generated based on the simulation setup, and the proposed feature extraction is implemented. Extracted cell-wise features are visualized. For example, two variables of interest (cumulative average intensity of detected points and cumulative intensity variance of detected points) are visualized with the ground truth labels of the obstacles for the scene with sparse vegetation (Figure 4.7). It can be observed that those variables contain informative information of the solid obstacles, as there is distinction between the cells containing cubes and those that do not contain cubes. More specifically, the darker spots in the middle of the plots (where the variables have the least values) are also where the cubes are located.

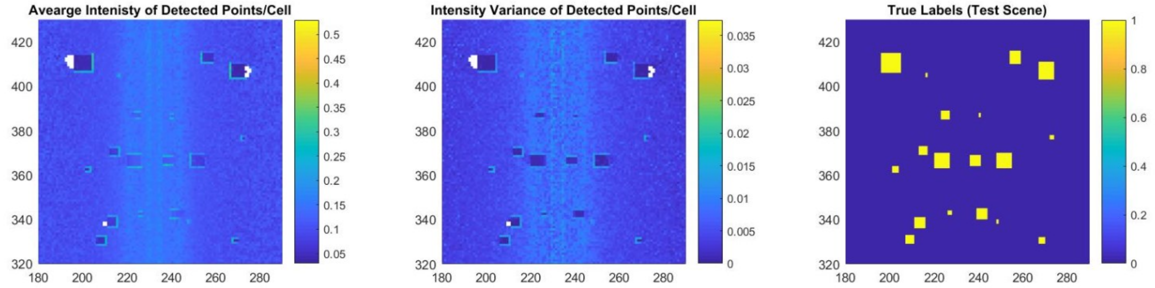


Figure 4.7 Plot of values for Average Intensity, Intensity Variance of detected points on the sparse scene compared to the location of true labels

Darker spots in the two plots on the left are where the cubes are located. The yellow spots in the right plot represents the true labels for the cells containing cubes.

According to the study design, data collected from training scenes were trained using two different classification algorithms. In the first step, no penalty term or fine tuning of the algorithms was applied. A 5-fold cross validation was used to evaluate the performance of algorithms on the training scenes. Table 4.2 lists confusion matrix, sensitivity, and accuracy values for the two classification algorithms applied on the training scene data with density level of 1. While both LDA and BT resulted in the same accuracy level, BT showed significantly better sensitivity results which is the main criterion to choose the final algorithm. The abbreviations used in the confusion matrix are defined as below:

— TN: True Negative- Classifier correctly identified a cell that is labeled 0 which means the cell contains no solid obstacle

— FN: False Negative- Classifier incorrectly identified a cell that is labeled 1 which means the cell contains solid obstacle, but the classifier predicted the label as 0

— FP: False Positive- Classifier incorrectly identified a cell that is labeled 0 which means the cell done NOT contain solid obstacle, but the classifier predicted the label as 1

— TP: True Positive- Classifier correctly identified a cell that is labeled 1 which means the cell contains solid obstacle

Table 4.2 Confusion Matrix, Sensitivity, and Accuracy results for the two classification algorithms used in the study

	Prediction/Reference				Sensitivity	Accuracy
	TN	FN	FP	TP		
LDA	9613	52	15	180	78%	99%
BT	9613	38	15	194	84%	99%

After Boosted Tree algorithm was chosen to proceed with, further steps were taken to fine tune the algorithm and optimize it based on the most important performance measure for this study: reducing false negatives. The cost function for the algorithm was adjusted to penalize false negatives seven times more than false positives. The penalty term was chosen based on several trial and errors to find a number that would improve sensitivity without too much adding to the false positives. Table 4.3 shows the comparison of Confusion Matrix, Sensitivity, and accuracy for the original algorithm (with no cost function) and the adjusted algorithm with the penalty term for false negatives which shows an improvement in the sensitivity value due to the added cost function to the classification algorithm.

Table 4.3 Confusion Matrix, Sensitivity, and Accuracy results for the original algorithm and the adjusted algorithm with penalty term for the sparse scene

	Prediction/Reference				Sensitivity	Accuracy
	TN	FN	FP	TP		
No Penalty	9613	38	15	194	84%	99.5%
Penalty	9607	24	21	208	90%	99.5%

The penalty improved Sensitivity by about 6% while also increasing the false positives by about 40%.

Figure 4.8 provides the comparison for eighty-four cumulative time frames per each density level. From the plot, it can be clearly interpreted that the algorithm almost always performs better on the lower density levels than the higher ones. The results, however, converge as more data frames get accumulated.

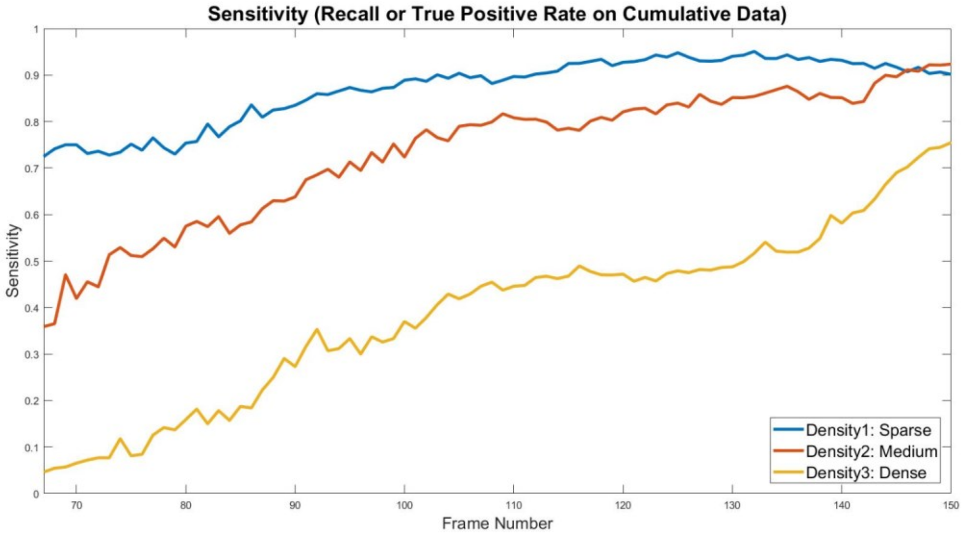


Figure 4.8 Comparison of Sensitivity results of the adjusted Boosted Trees algorithm given different vegetation density values

The average sensitivity rate for the sparse, medium, and dense scenes are 77%, 56% and 39% respectively.

Another way to get a visualization of the performance on different density levels is the use of different colors for each prediction category (TP, TN, FP, and FN). Figure 4.9 through 4.11 provide such visualization for the data frame 130. The reason data frame 130 is chosen for this visualization is that the vehicle is fairly in the middle of the scene on that time frame. Thus, the visualization can show the difference between the predicted labels for cells that are behind the vehicle and those that are ahead of it.

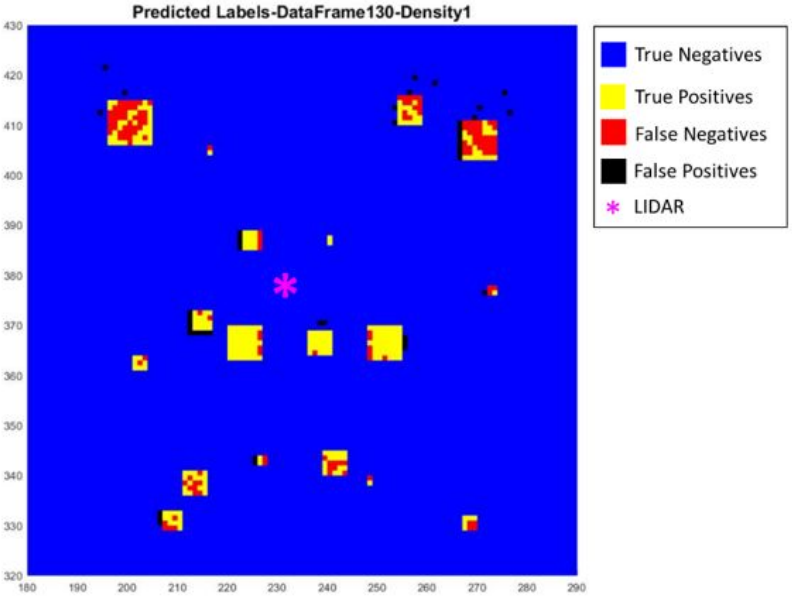


Figure 4.9 Predicted Labels on data frame 130 for the Sparse Scene

The magenta asterisk marks the location of the LIDAR when the data was captured. The vehicle started its path on a straight line from the middle of southernmost spot of the plot and ended on the middle of the northernmost spot of the plot

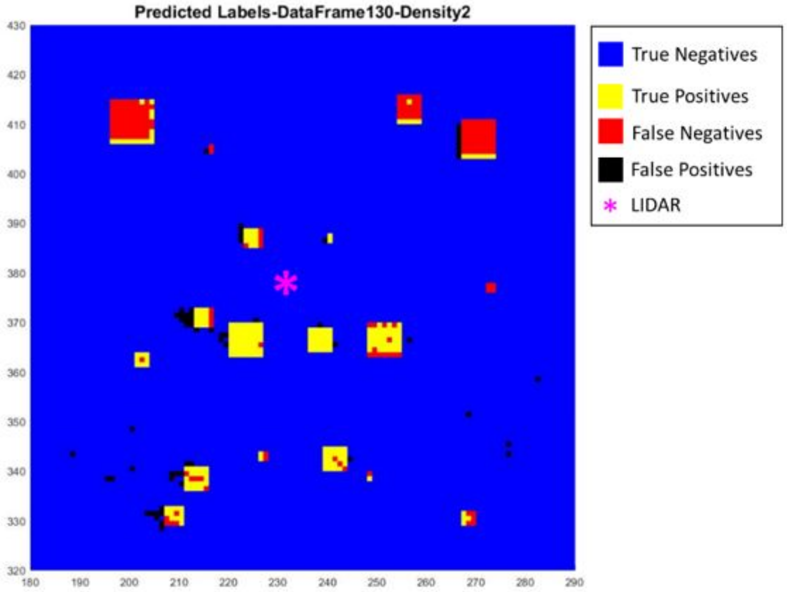


Figure 4.10 Predicted Labels on data frame 130 for the Medium Scene

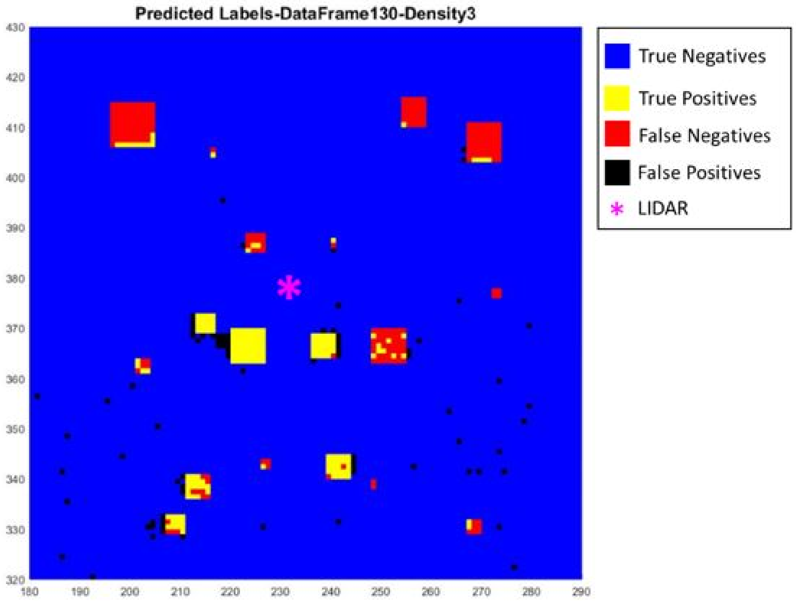


Figure 4.11 Predicted Labels on data frame 130 for the Dense Scene

From the Figure 4.9 through 4.11, it can be interpreted that the higher the density, the more false positives resulted by the classification algorithm.

The results of this study confirmed that increase in the density level of understory vegetation adversely affects the ability of classification algorithm to correctly detect solid obstacles. Using a cumulative data processing approach, however, the classification performance would improve and converge for different density levels. Between Boosted Trees (BT) and Linear Discriminant Analysis (LDA) algorithms, BT showed better performance with regard to sensitivity. Furthermore, a penalty term has been added to optimize sensitivity results. It is worth noting that the penalty term should be carefully chosen to improve the sensitivity. However, the penalty term should not significantly reduce the traversable area by excessive falsely classified cells. Plot of Sensitivity Results Over Time Frames (Figure 4.8) showed a consistent superiority of lower density levels over the denser ones. However, the results converge as more cumulative time frames data gets added to the test scenes.

One way that the sensitivity plots of different densities (Figure 4.8) can be used in decision making is by comparing the results with some predefined thresholds. For example, if it is decided that the autonomous operation may occur only under the condition that the sensitivity results are higher than 70%, then operations on density levels 2 and 3 should be avoided during path planning. Alternatively, operation on density Levels 2 and 3 may be allowed if we have somehow already accumulated sufficient data for higher density levels.

In addition, it can be observed that the increased density would increase the false positives. Even if we can keep the sensitivity results similar to the sparser scenes, increased false positives would limit the ability of vehicle to take the optimized route.

4.4 Conclusion

In this chapter, the impact of understory vegetation on the capability of solid obstacles identification based on LIDAR data is investigated. Cell-wise features are extracted from the entire gridded terrain, and classification algorithms, including Boosted Trees (BT) and Linear Discriminant Analysis (LDA) algorithms, are used for supervised learning. To deal with the imbalanced nature of the data, sensitivity is used to evaluate the performance of the classification methods.

Furthermore, in contrast to previous work that doesn’t provide quantitative details about the distribution of vegetation, the proposed method makes use of the physics-based simulation tool and defines the density levels in terms of number of plants that can be found in every square meter of the scene. This is a specific definition that is scalable and can be used for interpolation of detectability in other density levels.

A few questions are still open for further studies. For example, the simulated scenes used in evaluating the proposed method are completely flat, and future research opportunity exists to further examine the effect of understory vegetation in steep or hilly scenes. Additionally, the use of local accumulation, i.e. accumulation of certain recent time frames rather than cumulating all data frames along the way, might improve the overall computational performance of the model. Last but not least, the identification of solid obstacles with diversified geometries can also be investigated.

CHAPTER V

TRAVERSABILITY RISK MAPPING THROUGH UNDERSTORY DENSITY ESTIMATION FOR OFF-ROAD AUTONOMOUS GROUND VEHICLES

5.1 Introduction

Off-road autonomous driving systems pose more challenges than urban autonomous applications due to the high uncertainty of off-road environments. More specifically, the understory vegetation introduces a high level of uncertainty in obstacle detection, potentially leading to severe vehicle damage and even operator injury or death by occluding potential obstacles in the terrain. In this chapter, a data-driven model was proposed to estimate the density of understory vegetation to identify safe-to-navigate areas in the context of unstructured environments. Using a physics-based autonomous driving simulator to create vegetated scenes and generate navigation and LIDAR data, a group of grid-based features were proposed to characterize the spatial distribution of the point cloud in each cell, and a regression-based framework was proposed to estimate the understory density. Additionally, the impact of the cell size in the gridded x-y plain on the accuracy of the understory estimation was studied in this paper. The results showed that larger cell sizes would lead to higher adjusted R-squared values. Subsequently, by leveraging the optimized regression model, traversability risk maps were obtained to simultaneously identify both potential risks, dense understory vegetation, and solid obstacles. The proposed framework was validated by case studies of off-road autonomous navigation with both vegetation and solid obstacles in the scenes. This chapter contains the content of a paper submitted to *Journal of Autonomous Vehicles and Systems* in July 2020.

5.2 Materials and Methods

5.2.1 Simulation Setup

Sensor configuration and MAVS output data structure was identical to the work that was presented in the previous chapter. Also, similar to the previous work, scenes were designed to be 100 meters by 100 meters. Similarly, couch grass was used as the understory vegetation for the scenes. The height of selected understory plant ranged from 46 cm to 81 cm in the designed scenes with the average height of 64 cm. Six independent scenes were developed for this study. The first four were used to train and test different understory density estimation models and the fifth and sixth scenes were used for the risk map development purpose. The first set of scenes did not contain any solid obstacles. Each of those scenes was comprised of different segments (areas) of 25 meters by 25 meters with different density levels. The density of the understory in each segment of the scene was specified by the average number of couch grass plants located in each square meter of that segment. In addition, a straight pathway was generated in each terrain. The path started just before entering the vegetated scene and ended right after the vehicle completed traversing through the scene covered with understory vegetation. The first two scenes (Scene 1 and Scene 2) were used to fit the model and train the algorithm. Then, the fitted model was used to make predictions on the test scenes (Scene 3 and Scene 4). Test scenes were intentionally designed to be more complex than the first two scenes. That would enable assessment of the selected model's robustness in terms of its capability to be trained on simpler scenes but still be able to provide reliable estimates on the more complicated scenes with density levels that it had not been exposed to in the training step. Table 5.1 provides a summary of density levels -on a scale of 0 to 10- used in the first four scenes. A score of 0 denoted an area with no understory vegetation and 10 denoted an area with an average of 10 understory vegetation plants in each square meter. As can be seen in the Table 5.1, each scene had areas with varying density levels, simulating what one would encounter in the real world.

Similar to the first four scenes, Scene 5 and Scene 6 were also designed to cover square area of 100 meters by 100 meters. Similarly, couch grass plants covered the entire scenes. The understory plants were uniformly distributed across the entire scene with an average of 3 plants per each square meter for the fifth scene and 9 plants per each square meter for the sixth scene. Also, Scene 5 and Scene 6 consist of 18 cubes (solid obstacles) with different sizes. Figure 5.1 through 5.5 provide 2D illustrations of the scenes developed for this study.

Table 5.1 Scene designs with respect to vegetation density levels (Scene 1 through Scene 4)

	Density Level (Scale 0 to 10)										
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Scene 1	✓	✓				✓				✓	
Scene 2	✓			✓				✓			
Scene 3	✓	✓	✓	✓		✓		✓		✓	✓
Scene 4	✓	✓		✓	✓	✓		✓	✓	✓	

A black tick mark denotes a density level that was used in all four scenes. Green tick marks denote density levels common between Scene 1 and the test scenes (Scene 3&4). Blue tick marks denote density levels common between Scene 2 and the test scenes. Red tick marks denote density levels that were only used in Scene 3. Finally, purple tick marks denote density levels that were only used in Scene 4.

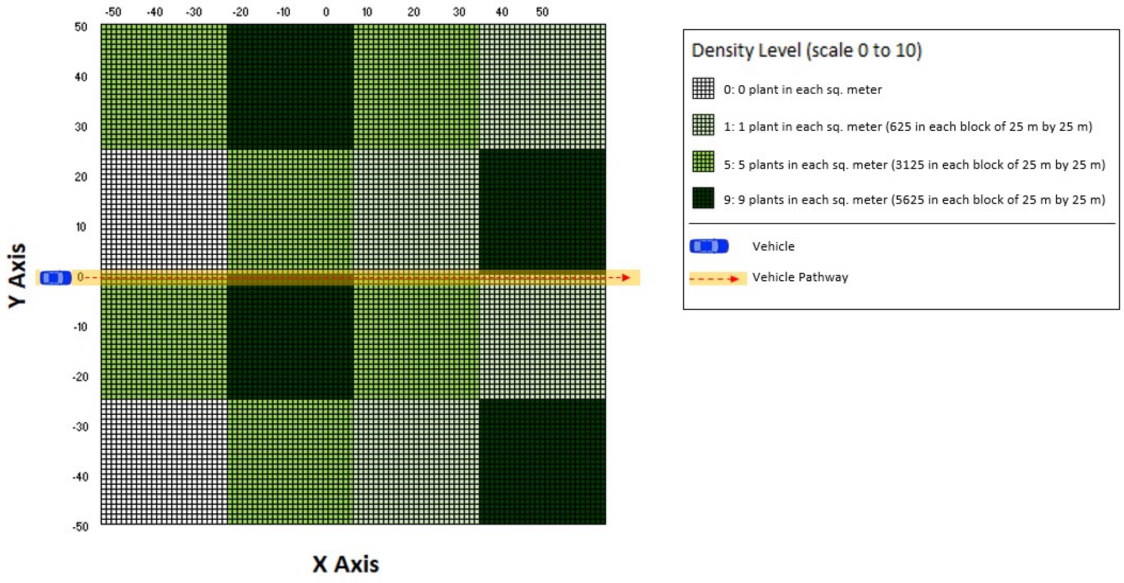

-50
-40
-30
-20
-10
0
10
20
30
40
50
Y Axis
X Axis
Density Level (scale 0 to 10)
0: 0 plant in each sq. meter
1: 1 plant in each sq. meter (625 in each block of 25 m by 25 m)
5: 5 plants in each sq. meter (3125 in each block of 25 m by 25 m)
9: 9 plants in each sq. meter (5625 in each block of 25 m by 25 m)
Vehicle
Vehicle Pathway

Figure 5.1 2D illustration of designed Scene 1

Each cell in the gridded space represents a 50 cm by 50 cm square.

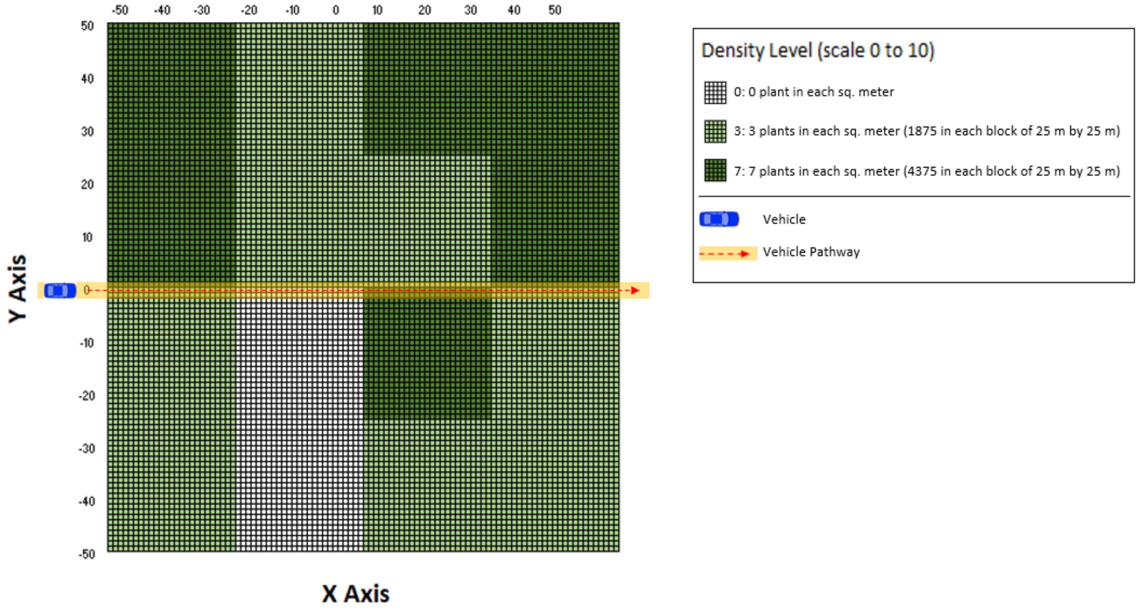

-50
-40
-30
-20
-10
0
10
20
30
40
50
Y Axis
X Axis
Density Level (scale 0 to 10)
0: 0 plant in each sq. meter
3: 3 plants in each sq. meter (1875 in each block of 25 m by 25 m)
7: 7 plants in each sq. meter (4375 in each block of 25 m by 25 m)
Vehicle
Vehicle Pathway

Figure 5.2 2D illustration of designed Scene 2

Location, size and density of vegetated areas in this scene differ from those of in scene 1

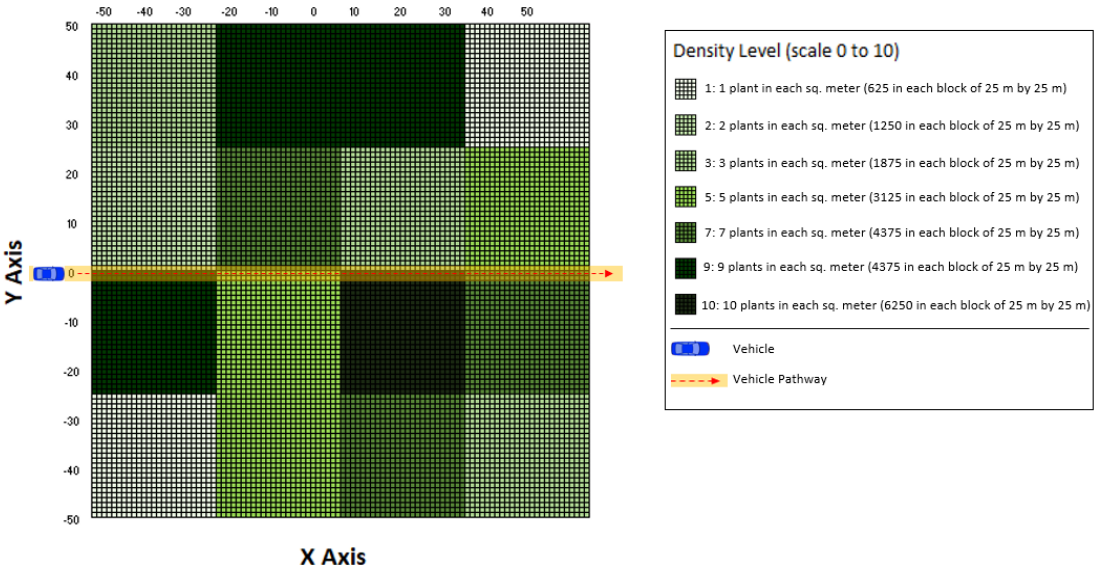

-50
-40
-30
-20
-10
0
10
20
30
40
50
Y Axis
X Axis
Density Level (scale 0 to 10)
1: 1 plant in each sq. meter (625 in each block of 25 m by 25 m)
2: 2 plants in each sq. meter (1250 in each block of 25 m by 25 m)
3: 3 plants in each sq. meter (1875 in each block of 25 m by 25 m)
5: 5 plants in each sq. meter (3125 in each block of 25 m by 25 m)
7: 7 plants in each sq. meter (4375 in each block of 25 m by 25 m)
9: 9 plants in each sq. meter (4375 in each block of 25 m by 25 m)
10: 10 plants in each sq. meter (6250 in each block of 25 m by 25 m)
Vehicle
Vehicle Pathway

Figure 5.3 2D illustration of designed Scene 3

Location, size and density of vegetated areas in this scene differ from those of in Scene 1 and Scene 2. There are areas with densities that do not exist in either of the previous scenes.

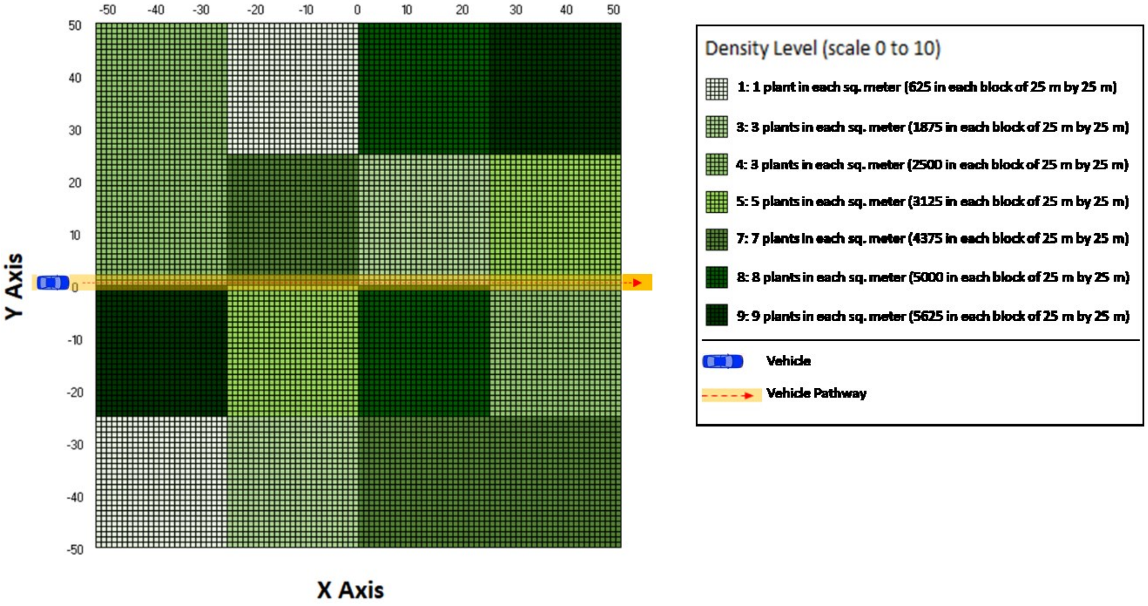

-50
-40
-30
-20
-10
0
10
20
30
40
50
Y Axis
X Axis
Density Level (scale 0 to 10)
1: 1 plant in each sq. meter (625 in each block of 25 m by 25 m)
3: 3 plants in each sq. meter (1875 in each block of 25 m by 25 m)
4: 3 plants in each sq. meter (2500 in each block of 25 m by 25 m)
5: 5 plants in each sq. meter (3125 in each block of 25 m by 25 m)
7: 7 plants In each sq. meter (4375 in each block of 25 m by 25 m)
8: 8 plants in each sq. meter (5000 in each block of 25 m by 25 m)
9: 9 plants in each sq. meter (5625 in each block of 25 m by 25 m)
Vehicle
Vehicle Pathway

Figure 5.4 2D illustration of designed Scene 4

There are areas with densities that do not exist in any of the previous scenes. This scene and Scene 3 will be used for testing the performance of the final algorithm.

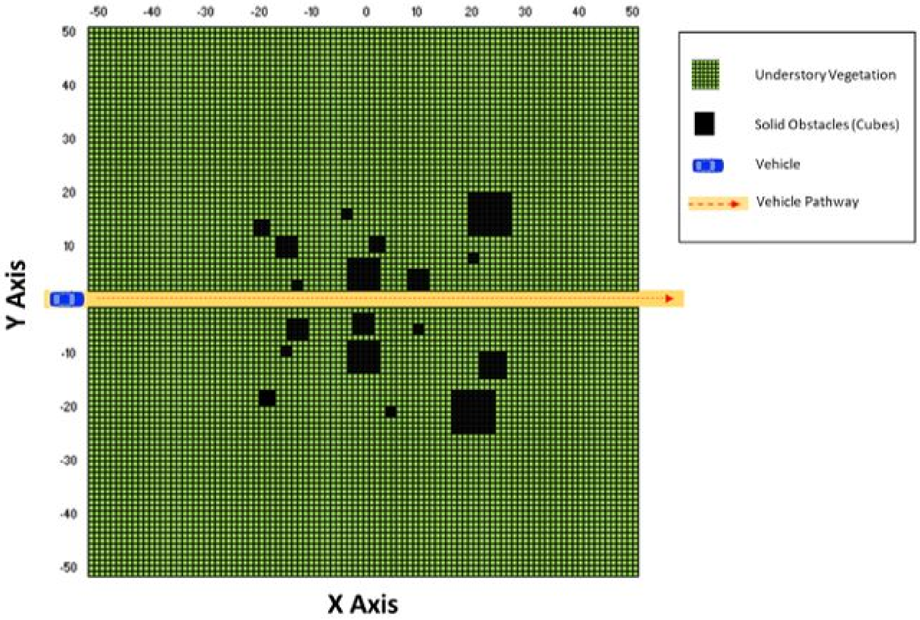


Figure 5.5 2D illustration of designed Scene 5 and Scene 6

The understory vegetation is uniformly distributed. 18 cubes are also located in the both sides of the pathway. Location of cubes are identical in both scenes. The density of the understory, however, differs between the two scenes with 3 plants/sq. m. for Scene 5 and 9 plants/sq. m. for Scene 6.

Figure 5.6 shows two examples of high-resolution rendering images captured from different viewpoints in Scene 1. From the pictures in Figure 5.6, different density areas in the scene are clearly distinguishable. As it was explained earlier, there is no solid obstacle in Scene 1.

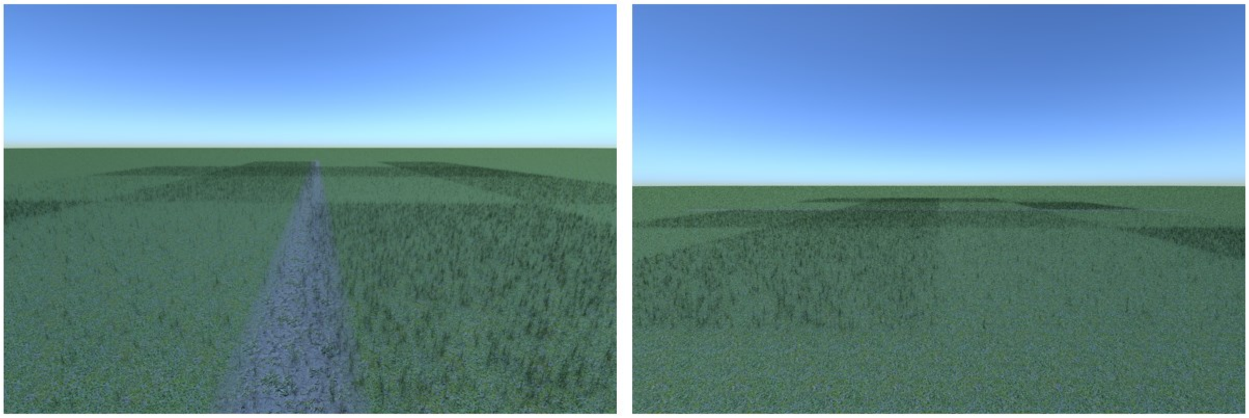


Figure 5.6 High-resolution rendering images from Scene 1

The pathway in the left scene is visible as a straight band dividing the scene to two segments on the left and hand side.

5.2.2 Proposed Methodology

5.2.2.1 Cell generation and feature extraction

This chapter used the same approach for the cell generation as described in CHAPTER IV, according to which each scene was first segmented into square cells in the x-y plane. Summary statistics were calculated for each cell in the gridded scene. Unlike the previous work, a cumulative feature extraction approach was not used in this chapter because in the current work, distance of each cell to the LIDAR was used as a feature. This feature could not be used with the cumulative approach as the distance might change in each timeframe and the values should be kept independently. Compared to the previous work, four entirely new predictor and one new response variable was added:

1. Distance from LIDAR (predictor)
2. Number of ground points (predictor)
3. Average number of detected points for the neighboring cells (predictor)
4. Average intensity of detected points for the neighboring cells (predictor)
5. Density (response variable)

Below are the steps taken to extract features from each cell at each time frame:

I. Number of points related features. Count the number of detected points *(N)* and number of ground points *(Ground_Count)* in each cell. Also, calculate the average number of detected points for the neighboring cells *(Neighbor_Mean[N])*.

I.a. Number of detected points for $C(i,j)$, denoted as $N_k(i,j)$ is the total number of point cloud records that fall in $C(i,j)$ for the *k*th data frame.

I.b. Number of ground points for $C(i,j)$, denoted as $Ground\_Count(i,j)$ is the count of those detected points considered to be part of the ground (and not a plant). $Ground\_Count(i,j)$ for the *k*th data frame can be calculated by subtracting those points within the *k*th data frame that have a height (z value) of larger than 0.005 meter from $N_k(i,j)$.

I.c. Average number of detected points for the neighboring cells for $C(i,j)$, denoted as *(Neighbor_Mean[N(i,j)])*, can be calculated for the *k*th data frame as:

$$\begin{aligned} &Neighbor\_Mean[N_k(i,j)] \\ &\qquad = [N_k(i,j+1) + N_k(i,j-1) + N_k(i+1,j) + N_k(i-1,j) \\ &\qquad + N_k(i+1,j+1) + N_k(i+1,j-1) + N_k(i-1,j+1) \\ &\qquad + N_k(i-1,j-1)]/8 \end{aligned} \tag{5.1}$$

II. Ray intensity related features. Calculate the mean and variance of ray intensity (μ_I and σ_I^2) for each cell. Also, calculate the average intensity of detected points for the neighboring cells *(Neighbor_Mean[I])*.

II.a. The mean of ray intensity for $C(i,j)$, denoted as $\mu_I\,(i,j)$, can be calculated for the *k*th data frame as:

$$\mu_{I_k}(i,j) = \sum I_K(i,j)/N_k(i,j) \tag{5.2}$$

where $\sum I_K(i,j)$ is the summation of all intensity values pertaining to $C(i,j)$.

II.b. The variance of ray intensity for $C(i,j)$, denoted as $\sigma_I^2(i,j)$, is calculated for the *k*th data frame as:

$$\sigma_{I_k}^2(i,j) = \frac{SS_{I_k}(i,j) - N_k(i,j) \times \mu_{I_k}(i,j)^2}{N_k(i,j)} \tag{5.3}$$

where $SS_{I_k}(i,j)$ is the sum of squares of intensity values pertaining to $C(i,j)$ for the kth data frame. $SS_{I_k}(i,j)$ is calculated as follow:

$$SS_{I_k}(i,j) = \sum_{n=1}^{n=N} I^2{}_{n_k}(i,j) \tag{5.4}$$

where $I^2{}_{n_k}(i,j)$ is the square of n^th^ intensity value pertaining to data located in $C(i,j)$ for the *k*th data frame.

II.c. Average intensity of detected points for the neighboring cells for $C(i,j)$, denoted as *(Neighbor_Mean[I(i,j)])*, can be calculated for the *k*th data frame as:

$$\begin{aligned} &Neighbor\_Mean[I_k(i,j)] \\ &\quad = [I_k(i,j+1) + I_k(i,j-1) + I_k(i+1,j) + I_k(i-1,j) \\ &\quad + I_k(i+1,j+1) + I_k(i+1,j-1) + I_k(i-1,j+1) \\ &\quad + I_k(i-1,j-1)]/8 \end{aligned} \tag{5.5}$$

III. Height related features. Calculate the mean and variance of Z values (μ_Z and σ_Z^2) for each cell. Also, identify the minimum and maximum z value (*Min Z* and *Max Z*) detected for each cell.

III.a. The mean of Z values for $C(i,j)$, denoted as $\mu_Z\,(i,j)$, can be calculated for the *kth* data frame as:

$$\mu_{Z_k}(i,j) = \sum Z_K(i,j)/N_k(i,j) \tag{5.6}$$

where $\sum Z_K(i,j)$ is the summation of all Z values pertaining to $C(i,j)$.

III.b. The variance of z values for $C(i,j)$, denoted as σ_Z^2 , can be calculated for the *k*th data frame as:

$$\sigma_{Z_k}^2(i,j) = \frac{SS_{Z_k}(i,j) - N_k(i,j) \times \mu_{Z_k}(i,j)^2}{N_k(i,j)} \tag{5.7}$$

where $SS_{Z_k}(i,j)$ is the sum of squares of z values pertaining to $C(i,j)$ for the kth data frame. $SS_{Z_k}(i,j)$ is calculated as follow:

$$SS_{Z_k}(i,j) = \sum_{n=1}^{n=N} Z^2{}_{n_k}(i,j) \tag{5.8}$$

where $Z^2{}_{n_k}(i,j)$ is the square of nth Z value pertaining to data located in $C(i,j)$ for the *k*th data frame.

IV. Distance related feature. Calculate the distance of each cell from the LIDAR. *Distance* for $C(i,j)$ can be calculated for the *k*th data frame as:

$$Distance_k(i,j) = \sqrt{[(V_{xCell_k} - i) \times Cell\_Size]^2 + [\left(V_{yCell_k} - j\right) \times Cell\_Size]^2} \tag{5.9}$$

where V_{xCell_k} and $V_{yCell_k}$are the x and y indices of the cell that the vehicle is located in the *k*th data frame. $Cell\_Size$ is the length of grid cells' side.

V. Density related feature. Calculate the density of understory vegetation, denoted as *Cell_Density,* for $C(i,j)$. *Cell_Density* for $C(i,j)$ can be calculated as:

$$Cell\_Density(i,j) = Plant\_Count(i,j)/Cell\_Size^2 \tag{5.10}$$

where $Plant\_Count$ is the number of understory vegetation plants located in $C(i,j)$. *Plant_Count* is a 'ground truth' value that is retrieved from the vegetation log data.

5.2.2.2 Understory density estimation and risk map development based on extracted features

The proposed methodology consisted of two phases. The first phase establishes the best regression model for understory density estimation. The second phase applies the density prediction model for real-time traversability mapping.

Phase I: Regression model selection for density estimation. Three regression methods that used different statistical approaches were implemented. The idea was to select the algorithm that provides "the best" prediction performance given the variables and nature of the data used for the study. The three regression algorithms were linear regression (with interactions and polynomial term), decision trees (simple trees method), ensemble of trees (boosted trees method).

To evaluate and compare the performance of the aforementioned regression methods, adjusted R-squared was used as the main measure instead of R-squared. R-squared will always improve as the number of variables increases, leading to an overfitted model. However, adjusted R-squared resolves this issue by adding a penalty term to the R-squared formula that penalizes more complicated models (Guanga 2019). Moreover, using the best regression model selected during this phase, the impact of changes in the size of each cell in the gridded scene on the accuracy of density estimation was studied.

Phase II: Terrain traversability mapping based on density prediction. The trained best model from the first phase was applied in real time to developing a traversability map based on real-world terrains with both understory and obstacles. In this context, a density threshold could be proposed as part of the predictive model for identifying cells containing solid obstacles. If the predicted density for the cell was higher than a certain threshold, the cell was deemed to contain a solid obstacle. This was based on the assumption that in the real world, each cell cannot contain more than a certain number of the understory plants due to the dimensions and biological needs of each type of plant. Thus, if the predicted density of a cell exceeded the defined threshold, it was highly likely that this cell contained a solid obstacle resulting high density estimation. For this study, an estimated density of 30 was used as the threshold. This number was chosen based on some trial and error practices and the fact that the average density level for the denser scene was 9 plants per each square meter which is well below the identified threshold.

Below is a summary of main steps of the proposed methodology for this chapter:

1. Set the *Cell_Size* to 50 cm and extract the features (predictors and response variable). The initial *Cell_Size* is consistent with the work in CHAPTER IV .
2. Consolidate data for all time frames of each scene in one table for model fitting purposes.
3. Eliminate records with $Distance > 30m$. This limitation was set to reduce the data size aiming to increase the processing speed while keeping enough data needed for the autonomous system to make proper navigation decisions regarding its immediate surroundings.
4. Use the three regression methods i.e. linear regression, simple decision trees, and boosted trees to train the data for Scene 1. Use eleven extracted features as described in 5.2.2.1 i.e. *N, Ground_Count, Neighbor_Mean[N],* $\mu_{I,}$ σ_I^2, *Neighbor_Mean[I],* $\mu_{Z,}$ σ_Z^2, $Min\ Z, Max\ Z$ and $Distance$ as predictors, and use *Cell_Density* as response variable (use 5-fold CV method).

5. Compare the performance of the regression algorithms and choose the best one (use 'adjusted R-squared as the main criterion).
6. Use the selected model from step 5 to fit the data for Scene 2 to capture model results such as adjusted R-squared.
7. Study the impact of changes in *Cell_Size* on model's performance by incrementing the *Cell_Size*, extracting the features, and capturing 5-fold cross validation results on Scene 1 and Scene 2. Capture the average adjusted R-squared and the average RSME for the two scenes on each *Cell_Size* increment. The results can provide insight for practical navigation uses in which a *Cell_Size* needed to be specified prior to feature extraction and prediction steps. Include the following *Cell_Size* values for this step: 50 cm, 70 cm, 90 cm, 110 cm, 130 cm, 150 cm. 170 cm, 190, and 200 cm. The *Cell_Size* values larger than 200 cm result in highly aggregated data which deemed to reduce the desirable level of scene understanding.
8. Fit the selected model in step 5 to the combined data from the training scenes (first two scenes). Use the smallest *Cell_Size* that led to average adjusted R-squared of equal or larger than 0.8 in step 7. If no model resulted in an average adjusted R-squared of equal or larger than 0.8, then use model tuning techniques, or extract new features to increase the performance of the model. Then go to step 7.
9. Make predictions with the fitted model from step 8 on both test scenes (Scene 3 and Scene 4) and capture the performance results.
10. Use the selected model from step 8 to estimate the density of each cell in Scene 5 and Scene 6.
11. Use a time frame that the vehicle is in the middle of the scene. Then, develop a risk map based on the estimated densities from step 10 on each test scene.

12. Use ‘estimated density’ = 30 as the threshold for solid objects, meaning that if the estimated density for a cell is equal or higher than 30, label that cell as ‘cube’.
13. Develop a confusion matrix and evaluate the accuracy and sensitivity of the model based on the results from step 12.
14. Capture the RSME of the scenes with and without the cubes.

Aforementioned steps can be summarized in the form of a flow chart diagram. Figure 6.7 provides such an illustration.

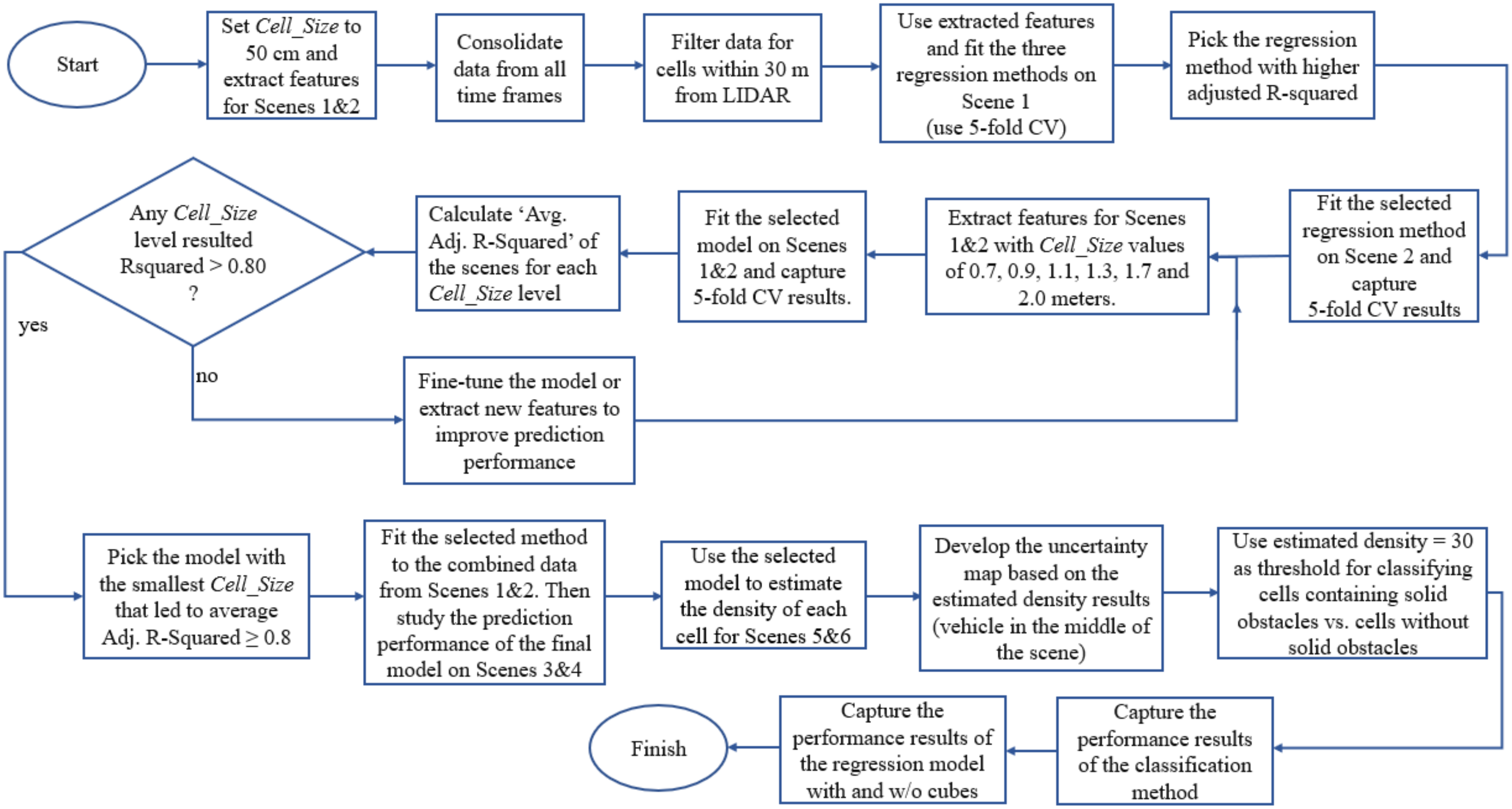

Start
Set *Cell_Size* to 50 cm and extract features for Scenes 1&2
Consolidate data from all time frames
Filter data for cells within 30 m from LIDAR
Use extracted features and fit the three regression methods on Scene 1 (use 5-fold CV)
Pick the regression method with higher adjusted R-squared
Fit the selected regression method on Scene 2 and capture 5-fold CV results
Extract features for Scenes 1&2 with *Cell_Size* values of 0.7, 0.9, 1.1, 1.3, 1.7 and 2.0 meters.
Fit the selected model on Scenes 1&2 and capture 5-fold CV results.
Calculate 'Avg. Adj. R-Squared' of the scenes for each *Cell_Size* level
Any *Cell_Size* level resulted Rsquared > 0.80 ?
yes
no
Fine-tune the model or extract new features to improve prediction performance
Pick the model with the smallest *Cell_Size* that led to average Adj. R-Squared ≥ 0.8
Fit the selected method to the combined data from Scenes 1&2. Then study the prediction performance of the final model on Scenes 3&4
Use the selected model to estimate the density of each cell for Scenes 5&6
Develop the uncertainty map based on the estimated density results (vehicle in the middle of the scene)
Use estimated density = 30 as threshold for classifying cells containing solid obstacles vs. cells without solid obstacles
Capture the performance results of the classification method
Capture the performance results of the regression model with and w/o cubes
Finish

Figure 5.7 Flowchart diagram of the proposed methodology for CHAPTER V

5.3 Results and Discussions

The scenes were generated based on the simulation setup, and the features were extracted according to the proposed gridding approach. Extracted cell-wise features were visualized to better understand the data and to visually validate the effectiveness of extracted features. For example, Figure 5.8 compares the understory density of each cell within Scene 1 with the concept design of Scene 1. According to the visualized data, the extracted feature (*Cell_Density*) correlated well with the actual density in the concept design.

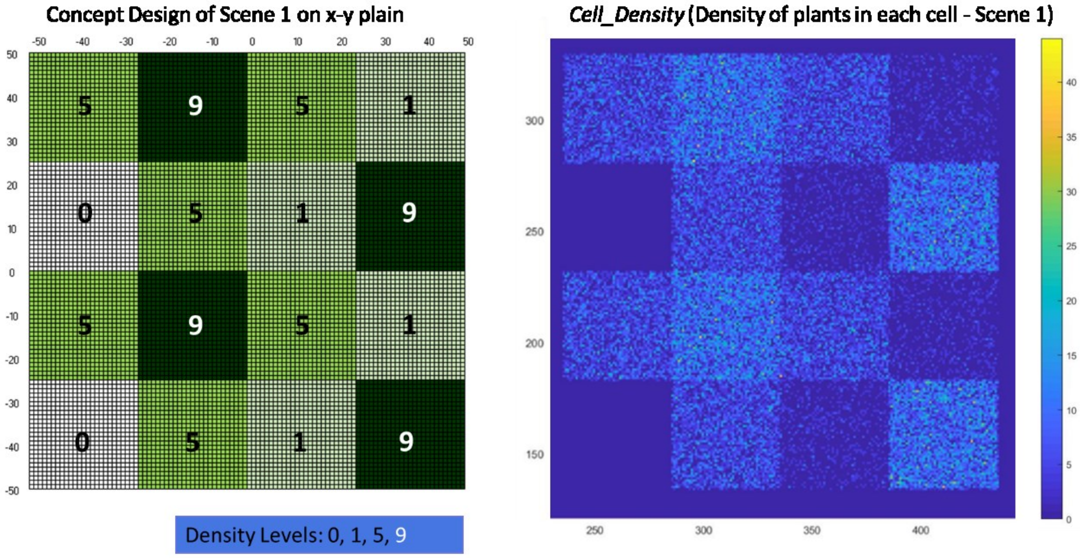


Figure 5.8 Comparison of concept Scene 1 with its cell-wise extracted density

The gridded scene in the left image contains cells with the side size of 50 cm. The numbers on the x-y axis represent the GPS spatial coordinates. The numbers on the x-y axis of the right image is the x-y index of each cell in the gridded area.

As another example, Figure 5.9 shows the average intensity and intensity variance of reflected lasers per each cell on a time frame that the vehicle is fairly in the middle of the scene. A closer look at each plot shows these two features are sensitive to the changes in the density levels. Furthermore, Figure 5.10 illustrates the correlation between the twelve extracted features including eleven predictors and one response variable. From the visual standpoint, not many highly correlated predictors can be identified in Figure 5.10. The only one that clearly stands out is the number of detected points (*N*) and the average of detected points in the neighboring cells (*Neighbor_Mean[N]*), because the density areas encompassed several neighboring cells. Thus, it was expected that the neighboring cells would follow similar pattern with regard to number of detected points.

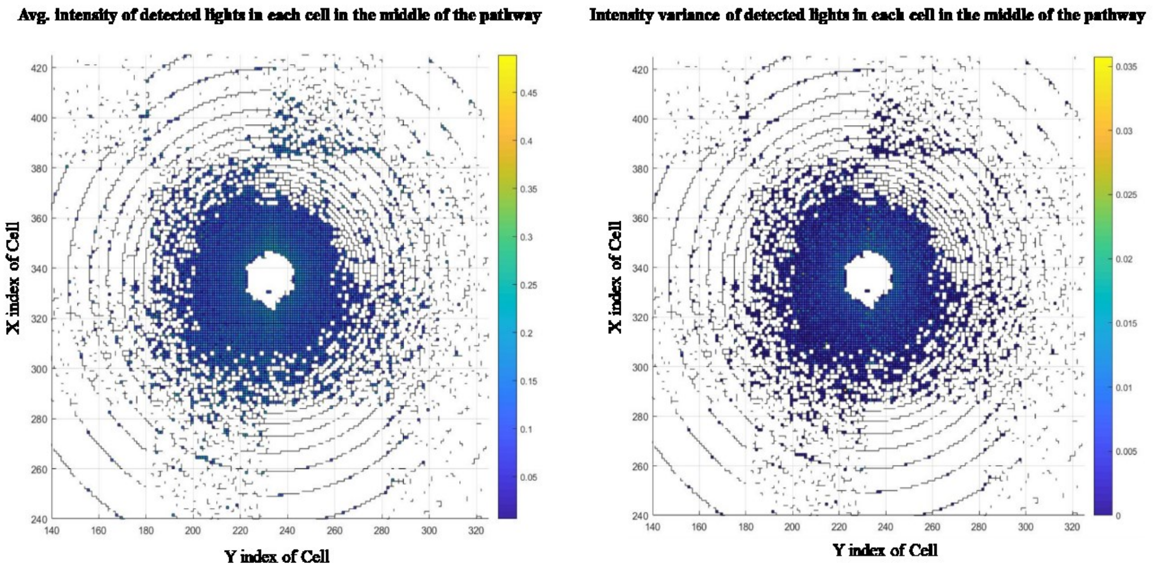


Figure 5.9 Plot of cell-wise average intensity and intensity variance on data frame 100.

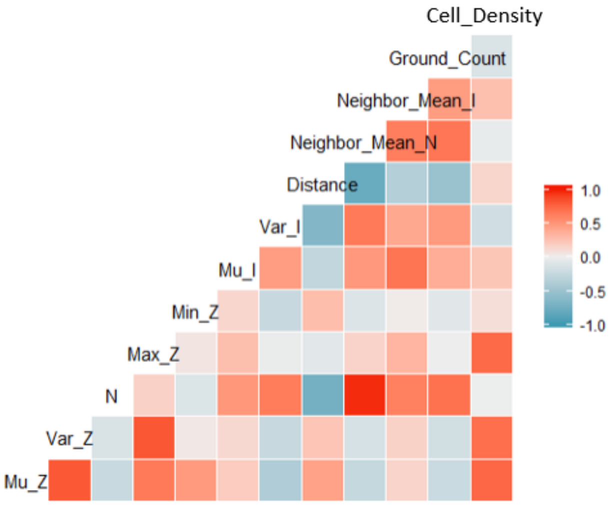


Figure 5.10 Correlation plot (Correlogram) of extracted features

Blue cells represent positive correlations whereas negative ones are displayed in red color. Color intensity is proportional to the correlation coefficients.

5.3.1 Regression model selection for density estimation

As part of the proposed methodology, features were extracted based on *Cell-Size* of 0.5 meter (i.e. cell area of 0.5 by 0.5 squared meter). Three different regression methods were used to fit the data. Table 5.2 provides a summary of the results. According to the proposed methodology, the adjusted R-squared was the main criterion to differentiate between the regression methods used in the study. As such the linear regression model in which quadratic interactions were included resulted in the highest adjusted R-squared. Table 5.2 provides a summary of the results for the three candidate algorithms used in this study.

Table 5.2 Summary of different algorithms' performance on Scene 1 with *Cell_Size* = 0.5 m

Algorithm	**RMSE**	**Adj. R^2**
Decision Trees (simple)	3.52	0.57
Ensemble of Trees (Boosted)	3.50	0.58
Linear Regression (w/ interactions)	3.40	0.60

Linear regression (w/ interactions) resulted the highest adjusted R-squared. The mean leaf size for decision trees and boosted trees are 6 and 8 respectively. Also, the number of learners for the boosted trees model is 30 with the learning rate of 0.1.

Assessing the importance of each variable (predictor) in a given predictive model can provide useful insight regarding the model. Figure 5.11 illustrates the Variables Importance Plot in which the interaction of Distance with other features is also considered. From the plot, it can be interpreted that three features had the highest importance (i.e. the highest effect) in correctly determining the cell density: 1) the average height of detected points, 2) the height variance of the detected points, and 3) the maximum detected height value for a given cell. This finding coupled with the learnings from the 'correlation plot' could be specifically instrumental in case further enhancement of the predictive model was needed. The results in Figure 5.11 can be explained by the fact that the scene only contained one type of vegetation. Thus, the intensity related features that in the existence of different object types could have higher importance, did not result in high values.

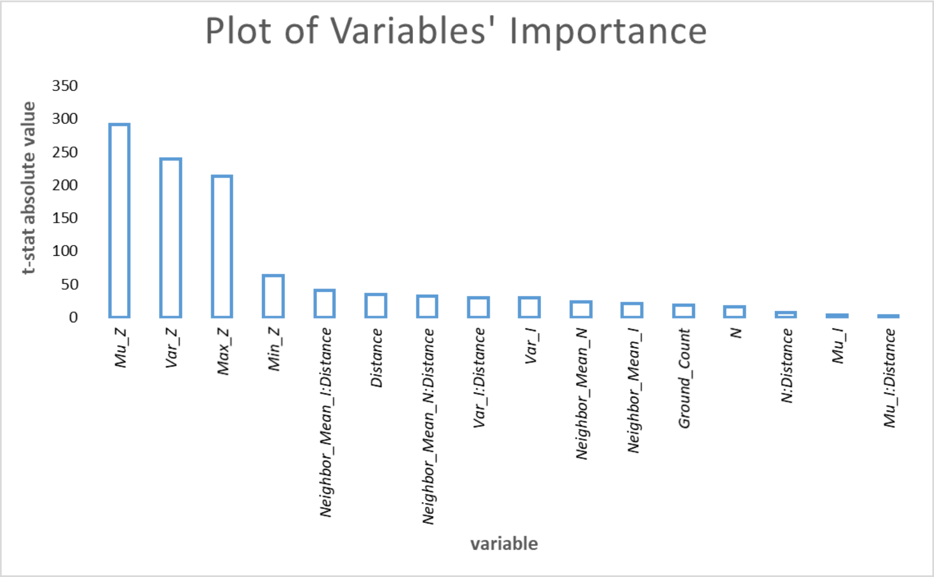


Figure 5.11 Plot of Variables' Importance (sorted)

The value of each bar represents the absolute value of the t-statistic for each model parameter (feature). These values are used as an indication of the parameter importance.

As part of this study, the effect of *Cell_Size* changes on the prediction accuracy of the selected model was investigated. This was deemed to aid in choosing the right *Cell_Size* based on predefined navigation requirements. Results of adjusted R-squared for the selected model (based on the predicted density for each cell) for all four scenes were captured in different *Cell_Size* levels. Table 5.3 provides a summary of the results for the first two scenes with regard to *Cell_Size* values starting from 50 cm and ending with 200 cm. The results showed that increase in the *Cell_Size* improved the adjusted R-squared values. Also, Figure 5.13 and Figure 5.13 provide a visual illustration of changes in average adjusted R-squared and average RSME for the first two scenes by different *Cell_Size* levels. From Figure 5.12, it can be interpreted that the average adjusted R-squared curve almost showed a linear relationship between the *Cell_Size* and the adjusted R-squared up to *Cell_Size* equal to 1.3 m. However, it flattened with larger levels of *Cell_Size* where the adjusted R-squared became closer to 1. Also, the average RMSE reduced with the increase of *Cell_Size* due the improved accuracy.

Table 5.3 Average and standard deviation of adjusted R-squared of the three scenes on different Cell_Size levels using Linear Regression (w/ interactions).

Cell Size (m)	**Scene1**	**Scene2**	**Average**	**Std. Dev.**
0.5	0.60	0.56	**0.58**	**0.019**
0.7	0.67	0.62	**0.64**	**0.024**
0.9	0.73	0.67	**0.70**	**0.029**
1.1	0.78	0.72	**0.75**	**0.029**
1.3	0.83	0.78	**0.81**	**0.025**
1.5	0.87	0.83	**0.85**	**0.019**
1.7	0.89	0.86	**0.88**	**0.016**
1.9	0.91	0.88	**0.90**	**0.011**
2.0	0.92	0.89	**0.90**	**0.011**

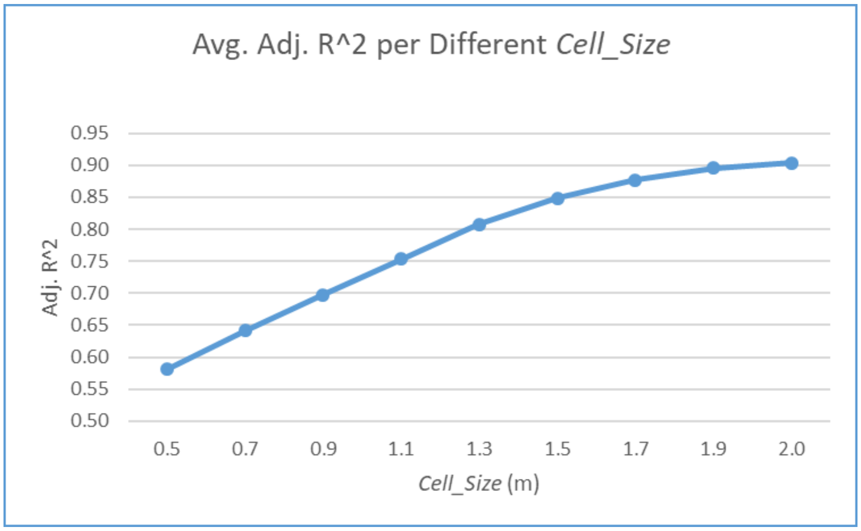


Figure 5.12 Graph of changes in average adjusted R-squared by different Cell_Size levels

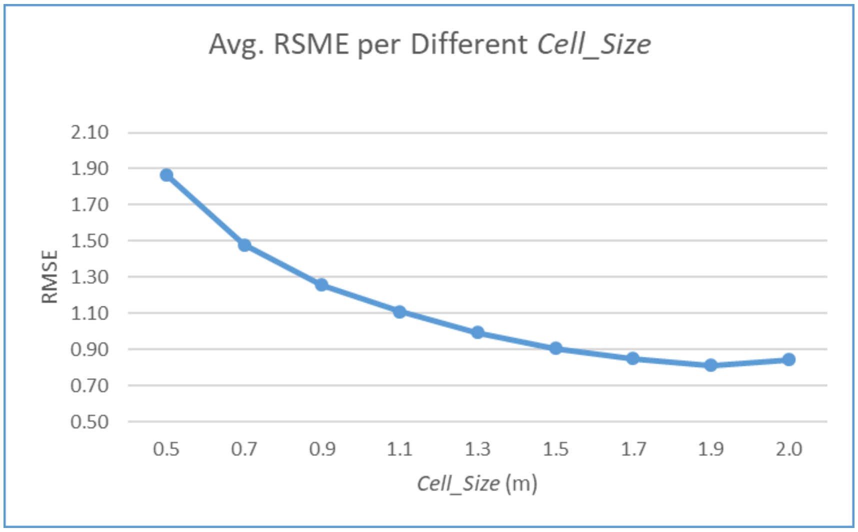


Figure 5.13 Graph of changes in average adjusted RMSE by different *Cell_Size* levels

The selected model (with *Cell_Size* = 1.3 m) was fitted to the combined data from the training scenes (scenes 3 and 4) and predictive performance was tested using all timeframes. Both tests resulted in adjusted R-squared of 0.81. Figure 5.14 and Figure 5.15 illustrate the spatial residual plots of predicted density for the understory vegetation of Scene 3 and Scene 4. The plots are developed based on a single frame where the vehicle is approximately in the middle of the scene. From the plots it can be interpreted that the performance of the density estimation algorithm varied across the scene. That could be explained by the fact that the scene encompassed areas with different density levels and the algorithm did not perform identically on different density levels. Furthermore, the farther the cell was from the LIDAR, the larger the absolute error was, because the closer cells could obtain more data points from the LIDAR and thus providing more precise characterization of the cell. The average error values (RMSE) for Scene 3 and 4 were 1.86 and 1.72 respectively.

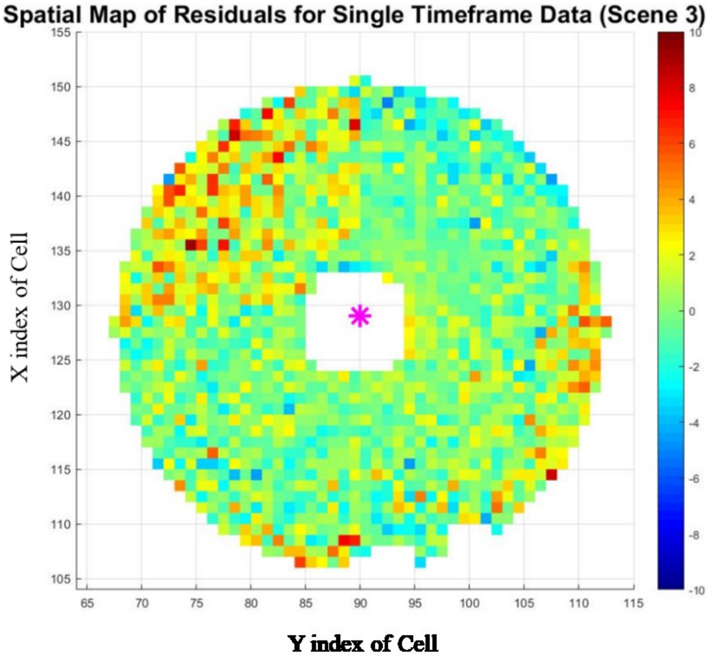


Figure 5.14 Spatial plot of residuals for Scene 3 (Single Timeframe)

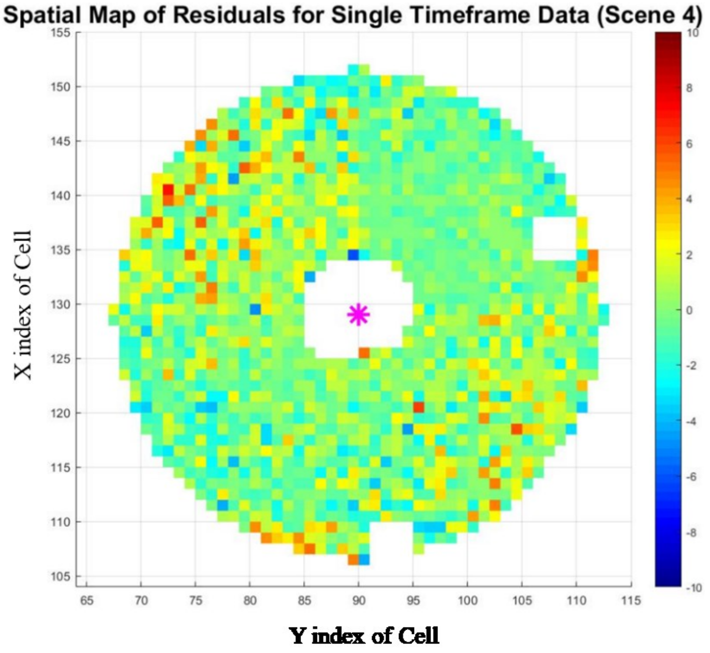


Figure 5.15 Spatial plot of residuals for Scene 4 (Single Timeframe)

5.3.2 Terrain traversability mapping based on density prediction

To develop the traversability maps, the final model (i.e. linear regression with the *Cell_Size* of 1.3 m) was used. Scenes 5 and 6 treated as a real-world situation in which there is no prior knowledge of the location of the cubes, nor there is prior knowledge with regard to the density of the overall scene or the segments (cells) within the scene. To make predictions, one timeframe was used when the vehicle was approximately in the middle of the scene. Figure 5.16 and Figure 5.17 illustrate the resulting traversability risk maps for the two scenes. The white areas are the areas for which there is no data captured in that specific time frame. Lack of data could be a result of LIDAR range or occlusion caused by obstacles in the scene. A yellow-green scale colormap represents the estimated density, hence the traversability risk. The darker an area, the higher the estimated density is. Using a density threshold of 30 plants/sq. m., predicted understory density was used to classify cells that contain the cubes. The thresholding also unifies the color of those cells that have densities higher than the specified threshold.

This representation can address two different risks: 1) risk of high understory vegetation, and 2) risk of solid obstacles. The resulting traversability risk maps clearly show that based on the understory vegetation density prediction, the risky regions in the terrain can be identified. Based on real-time collected point cloud data, those regions are regarded as either very densely vegetated areas or solid obstacles, which should be both avoided in path planning. In the first case, they should be avoided because they could reduce the obstacle detection accuracy. In the second case, they should be avoided as they pose imminent collision risk due to the existence of solid obstacles.

Table 5.4 provides the classification results for the obstacle detection as well as the regression results for the density estimation of the same timeframe that the risk maps were developed based upon.

Table 5.4 Prediction Results for Scene 5 and Scene 6 (Single Timeframe)

	Obstacle Detection						**Density Prediction**
	Prediction/Reference				**Sensitivity**	**Accuracy**	**RMSE**
	TN	**FN**	**FP**	**TP**			
Scene 5	1,435	2	28	13	87%	98%	1.29
Scene 6	1,506	18	66	23	56%	95%	2.25

Confusion Matrix, Sensitivity, and Accuracy results for the obstacle detection algorithm as well as RSME results of the vegetation density prediction algorithm. TN, FN, FP and TP represent 'True Negative', 'False Negative', 'False Positive', and 'True Positive' respectively.

From Figure 5.16 and Figure 5.17, it can be easily interpreted that compared to Scene 6, the traversability risk is lower in Scene 5 as the overall color is more inclined towards the lower end of the spectrum (yellow) while in the Scene 6 the overall scene is darker and is more toward the middle of the spectrum (green). Also, in line with the findings in CHAPTER IV, higher density resulted in an increase in misclassified instances, as illustrated by lower sensitivity value in Table 5.4 for Scene 6 as well as more black cells which can be regarded as false alarms in Figure 5.17.

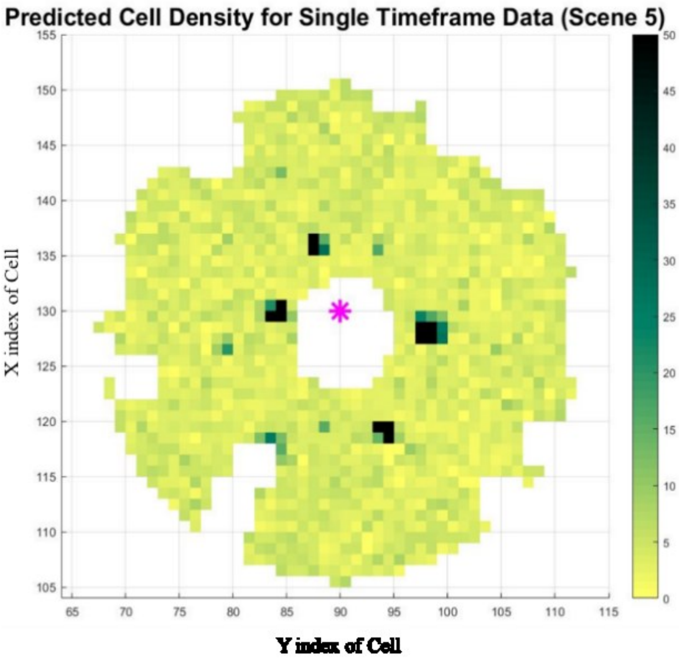


Figure 5.16 Traversability Risk Map: Density Prediction on Single Timeframe for Scene 5

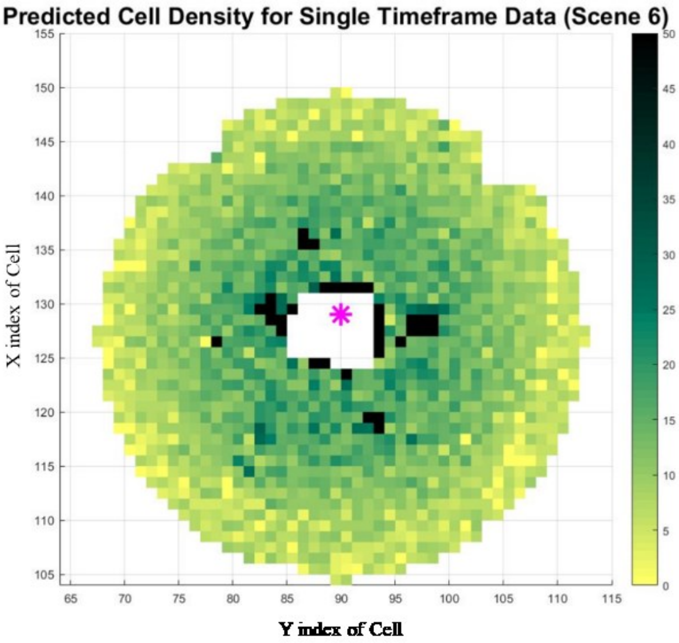


Figure 5.17 Traversability Risk Map: Density Prediction on Single Timeframe for Scene 6

5.4 Conclusion

As shown in the Chapter VI, understory density significantly affects the obstacle detection performance in scene understanding of off-road autonomous vehicle. In this chapter, a regression-based framework is proposed to use real-time LIDAR data to estimate understory vegetation density for assessing traversabilty risk in off-road autonomous navigation applications. Cell-wise features were extracted from the entire gridded terrain, and regression algorithms, including linear regression, decision trees, and ensemble of trees were used for supervised learning. Adjusted R-squared was used as the main metric to evaluate models' capability of explaining the variability. Results of this study show that using linear regression with interactions and squared terms outperforms other algorithms used in the study such as simple decision trees, and boosted trees. Also, the impact of *Cell_Size* changes on model's prediction accuracy was studied. It was shown that increase in the *Cell_Size* improved adjusted R-squared values. This improvement in adjusted R-squared values can be explained by the fact that an increase in the *Cell-Size* results in more aggregated data, which in turn results in a more robust characterization of each cell. Considering that a lot of the extracted features of this study were generated by calculating the average value of certain datapoints, a higher level of aggregation led to less noisy features extracted for both input and output variables. Generalizing the calculated summary statistics by enlarging the cells could increase the collision risk. The increased risk is due to the fact that the effect of LIDAR data reflected from obstacles within a cell weakens in the summary statistics as more data from non-obstacle objects are brought into the calculations. Therefore, the *Cell_Size* should be carefully chosen so that while the predictive models stay in high adjusted R-squared values, the impact of data from hazardous obstacles does not get buried under the load of other datapoints.

It was shown that given the regression model for understory density prediction and a predefined threshold for predicted density, traversability risk mapping can be obtained in real time by identifying both densely vegetated region and solid obstacles. Two testing scenes were used to validate the accuracy of proposed traversability risk mapping framework in terms of both RMSE of regression and obstacle identification accuracy. The results showed that the developed algorithm was able to reliably identify densely vegetated areas as well as areas that contained solid obstacles. While such type of study was not seen in the prior work, it was deemed to be essential for the path planning algorithms to be able to quantify the traversability risk and to make decisions based on the predefined detectability certainly level. Also, in line with the results obtained from the prior study (CHAPTER IV), it was verified that the accuracy of the obstacle detection algorithm was reduced as the understory density increased. This finding is specifically important from the safe navigation standpoint and can be used where a certain level of accuracy is required for autonomous operations.

A few questions, however, are still open for future studies. As an example, the scenes used in this study are completely flat which does not very well represent different scenarios that can be encountered by AGVs in the real world. Future research opportunities exist to adjust the current framework to achieve satisfactory density prediction results in steep or hilly scenes. Additionally, this study only used one type of understory vegetation i.e. couch grass. As an extension to this study, the understory vegetation density with diversified understory plants can be investigated. Also, this study was simulated with the weather parameters set to clear weather condition. Hence, the impact of severe weather conditions on the reliability of understory density estimation as well as detectability of solid obstacles should be studied.

CHAPTER VI

IDENTIFICATION OF TREES IN AN OBSTACLE-RICH FORESTED ENVIRONMENT FOR OFF-ROAD AUTONOMOUS GROUND VEHICLES

6.1 Introduction

In off-road environments, accurate classification of different obstacle types can significantly improve the path planning decision making for off-road autonomous systems. More specifically, in the context of forested environments, the ability of the scene perception algorithm to correctly distinguish trees from solid obstacles is of special importance. Classification of trees as solid obstacles would unnecessarily limit the traversable area. While tree trunks, in many cases, can be correctly classified as solid obstacles, tree crowns including leaves and thin branches might not be a collision hazard for navigation. From the safe navigation standpoint, the ability of making distinction between safe parts and non-safe parts of a tree can help the scene perception algorithm remain on a reliable level while improving the optimized path. To achieve the goal, the first step is developing a solution to correctly distinguishing trees from solid obstacles. This chapter proposes a new framework for such a solution. Relevant features were extracted from the scene data generated by MAVS and different classification models are used to investigate the effectiveness of the extracted features.

6.2 Materials and Methods

6.2.1 Simulation Setup

Three different scenes were designed to collect data and to test the predictive models. Similar to the previous chapters, scenes were designed to be 100 meters by 100 meters. While the focus of this study was on trees and solid obstacles (cubes), understory vegetation was added to the scenes to better represent the real situations. Similar to previous chapters (CHAPTER IV and V), couch grass was used as the understory vegetation. Also, the same sensor configuration was used for the simulation step as those used in the previous studies in this dissertation. For the purpose of this chapter, all scenes were designed to have the same type of tree and same physical properties related to cubes. However, the number of cubes, trees, their locations, and their average sizes (length, width, and height) differed scene to scene. The idea was to have random sizes as well as random orientation of the obstacles (trees and cubes) to better represent the real forested scenes.

Figure 6.1 provides an image of the tree used in the scenes. This tree belongs to the oak family. It is regarded as a good choice of tree as oak trees is one of the most widely spread trees on earth (Ward 2019).

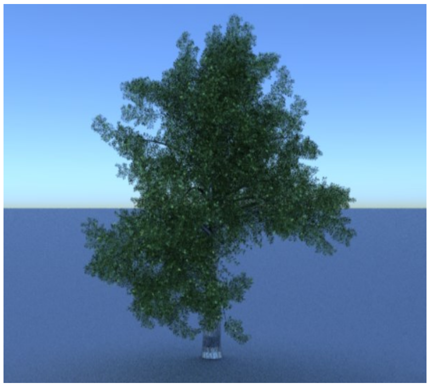


Figure 6.1 A tree from the oak family was used to develop the forested area

Figure 6.2 through 6.4 represent the ground truth information regarding the location of trees and cubes on each scene. Table 6.1 provides details about the height, length and width of obstacles in each scene. As for the trees, length and width of their bounding box was used to represent the area they cover on the x-y plane. In summary, the cubes were designed to be shorter than the trees to represent solid obstacles that could partly or entirely become occluded by the understory vegetation. Also, in order to diversify the scenarios and avoid data bias, Scene 1 was designed to have the same number of trees and cubes while Scene 2 had more trees than cubes, and Scene 3 had more cubes than trees.

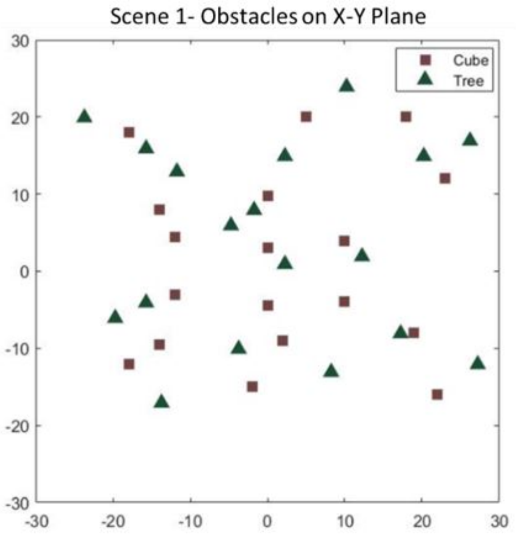


Figure 6.2 Scene 1: Location of Trees and Cubes

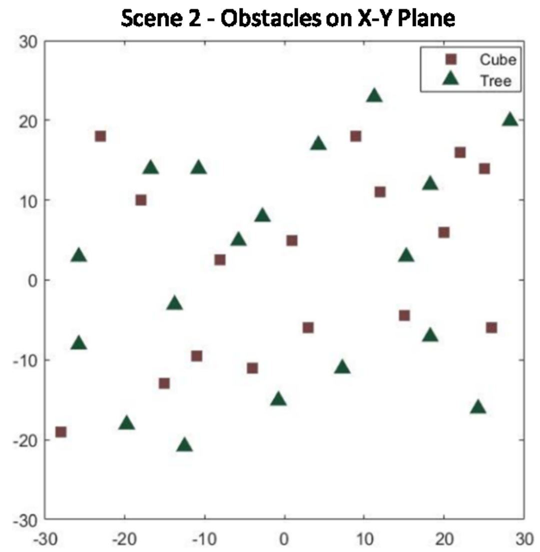


Figure 6.3 Scene 2: Location of Trees and Cubes

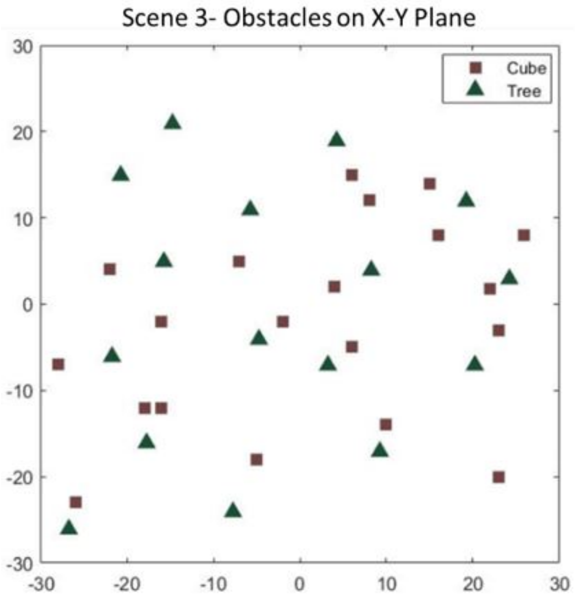


Figure 6.4 Scene 3: Location of Trees and Cubes

Table 6.1 Properties of Objects in Designed Scenes

	# of Trees	# of Cubes	Understory Density (plants/m^2)	Mean Tree Height (m)	Mean Tree BB Length (m)	Mean Tree BB Width (m)	Mean Cube Height (m)	Mean Cube Length (m)	Mean Cube Width (m)
Scene1	18	18	5	6.37	4.82	5.1	0.89	1.87	1.87
Scene2	18	16	5	6.22	5.78	5.84	0.98	2.06	2.06
Scene3	17	20	5	6.36	6.25	6.28	0.90	1.55	1.55

Figure 6.5 shows a high-resolution rendering of Scene 1 captured in a time frame that the vehicle was passing through the scene. Trees, cubes, and understory vegetation can be visually distinguished with the camera view. However, the challenge that this study was focused on was to make reliable distinctions solely based on the LIDAR data and vehicle's positions.

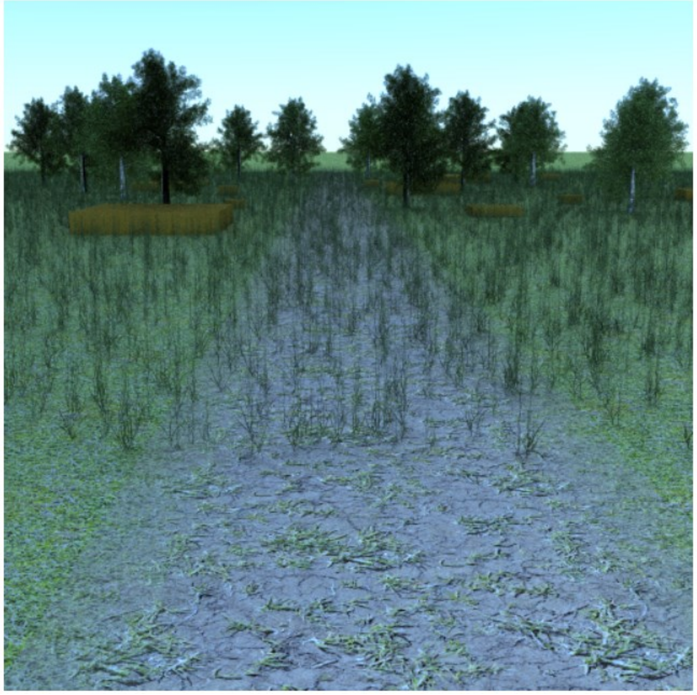


Figure 6.5 Scene 1 from a single time frame camera viewpoint

6.2.2 Dataset Description

Data files used in this study were similar to those in CHAPTER V. Also, dataset structures were identical. However, the bounding box dimensions of the cubes and trees from the Ecosystem Objects File (APPENDIX C) that was not used in the previous study, was used in this chapter. The bounding box info was needed in the cell generation and feature extraction step.

6.2.3 Proposed Methodology

6.2.3.1 Cell generation and feature extraction

All the three studies in this dissertation have a lot of commonality in their approaches as they are intended to fit in the same bigger theme i.e. traversability analysis in unstructured forested terrains for off-road autonomy using LIDAR data. Part of such commonality is seen in the cell generation and in the cell-wise feature extraction. This chapter uses exactly the same approach for the cell generation as was used in the previous two studies covered in this dissertation according to which each scene is first segmented into square cells in the x-y plane. Various summary statistics are calculated for each cell in the gridded scene. Nine out of twelve summary statistics calculated for this study are shared with the pervious study in CHAPTER V, including number of detected points *(N)*, average number of detected points for the neighboring cells *(Neighbor_Mean[N])*, mean and variance of ray intensity (μ_I and σ_I^2), mean and variance of Z values (μ_Z and σ_Z^2), minimum and maximum z value (*Min Z* and *Max Z*), and distance of each cell from the LIDAR *(Distance)*. However, each study has its unique cell-wise features that specifically address the problem related to the respective study. Regarding the topic of this chapter, three new features were extracted that were deemed to be useful in increasing the performance of the proposed model for tree vs. solid obstacle detection. The three features are:

I. **Average height of detected points for the neighboring cells**: Average height of detected points for the neighboring cells for *C(i,j)*, denoted as *Neighbor_Mean[Z (i,j)],* can be calculated for the kth data frame as:

$$
\begin{aligned}
&Neighbor\_Mean[Z_k(i,j)] \\
&\qquad = [\mu_{Z_k}(i,j+1) + \mu_{Z_k}(i,j-1) + \mu_{Z_k}(i+1,j) + \mu_{Z_k}(i-1,j) \\
&\qquad + \mu_{Z_k}(i+1,j+1) + \mu_{Z_k}(i+1,j-1) + \mu_{Z_k}(i-1,j+1) \\
&\qquad + \mu_{Z_k}(i-1,j-1)]/8
\end{aligned}
\tag{6.1}
$$

II. **Number of detected points at the breast height**: According to Nix (2019), “Breast height is specifically defined as a point around the trunk at 4.5 feet (1.37 meters in metric using countries) above the forest floor on the uphill side of the tree”. While 1.37 meters is mainly used in the US as the breast height (USDA Forest Service 2015), 1.3 meters (about 4.3 feet) is used as the breast height in many countries as well as in many studies such as those in McDaniel et al. (2012); Feldpausch et al. (2011); Maas et al. (2008); Véga et al. (2016). For the purpose of this study and in order to enable collecting more data, a height range of 30 cm is used as an indication of the breast height, starting from 1.30 meters above the forest floor up to 1.60 meters above the forest floor on the uphill side of the tree. Then, the previously extracted feature of number of detected points (*N*) was adjusted to only include those points fall in the defined breast height range. Therefore, number of detected points at the breast height for *C(i,j)*, denoted *as N_BH (i,j)*, can be calculated for the kth data frame as:

$$N\_BH_k(i,j) = \sum n_k(i,j) \qquad given\ \ 1.30\ m < Z_n < 1.60\ m, \qquad and\ \ n \in C(i,j) \tag{6.2}$$

where Z_n is the height of point *n* that belongs to *C(i,j)*.

III. **Average intensity of detected points below the breast height**: Average intensity of detected points below the breast height for the neighboring cells for *C(i,j)*, denoted as μ_{I_UBH}*(i,j)*, can be calculated for the kth data frame as:

$$\mu_{I\_UBH_k}(i,j) = \sum I\_UBH_K(i,j)/N\_UBH_k(i,j) \tag{6.3}$$

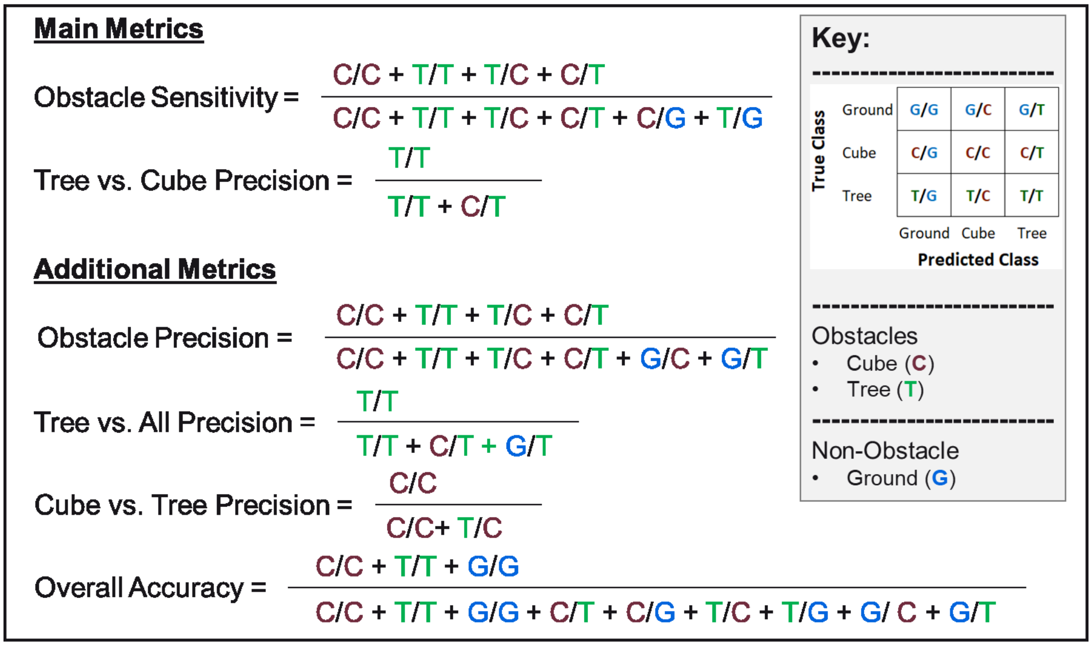
where $\sum I\_UBH_K(i,j)$ is the summation of all intensity values pertaining to $C(i,j)$ that have z values (height) lower than 1.30 meters. Also, $N\_UBH_k(i,j)$ is the count of all points pertaining to $C(i,j)$ with height lower than 1.30 meters.

6.2.3.2 Identification of trees in obstacle-rich off-road scene

Thirteen classification models that fall under four main classification categories of Discriminant Analysis (DA), Nearest Neighbor Classifiers, Decision Trees, and Ensemble Classifiers were implemented. The idea was to select and proceed with the model that provides better performance given the study data and settings. Those thirteen classification models are: Simple Decision Tree, Medium Decision Tree, Complex Decision Tree, Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Ensemble Classifier (Subspace DA), Ensemble Classifier (Boosted Trees), Ensemble Classifier (Bagged Trees), Ensemble Classifier (Subspace KNN), KNN (Fine), KNN (Medium), KNN (Coarse), and KNN (Cosine), To analyze the performance of classification models that were considered for this study, six model assessment metrics were used. Two out of those are more important due to their higher relevancy to the topic of this study. Those two metrics are:

1- **Obstacle Sensitivity**: In the context of this chapter, both trees and cubes are considered obstacles. This criterion is in line with CHAPTER IV of this dissertation in which the focus was on detecting the obstacles in the scene. In this context, sensitivity is an important measure as it addresses safe traversabilty concerns. The reason this criterion has been used in this study was explained in more details in CHAPTER IV. This criterion is used for the initial model selection.

2- **Tree vs. Obstacle Precision**: This criterion addresses the main purpose of this chapter (i.e. identification of trees in an obstacle-rich forested environment for off-road AGVs). Precision (Positive Predictive Value) was selected because the developed algorithm was intended to identify as many trees as possible without an elevated false positive rate which in this context meant cubes that were classified as trees. Such an intent was best measured by calculating Tree vs. Cube Precision. It is worth noting that optimizing for recall instead of precision in this context would unnecessarily increase the false positive rate. Figure 6.6 provides the formulas for the two main metrics as well as the additional metrics that were used to investigate the performance of classification models for this study.



Main Metrics

$$\text{Obstacle Sensitivity} = \frac{C/C + T/T + T/C + C/T}{C/C + T/T + T/C + C/T + C/G + T/G}$$

$$\text{Tree vs. Cube Precision} = \frac{T/T}{T/T + C/T}$$

Additional Metrics

$$\text{Obstacle Precision} = \frac{C/C + T/T + T/C + C/T}{C/C + T/T + T/C + C/T + G/C + G/T}$$

$$\text{Tree vs. All Precision} = \frac{T/T}{T/T + C/T + G/T}$$

$$\text{Cube vs. Tree Precision} = \frac{C/C}{C/C + T/C}$$

$$\text{Overall Accuracy} = \frac{C/C + T/T + G/G}{C/C + T/T + G/G + C/T + C/G + T/C + T/G + G/C + G/T}$$

Key:

True Class	Ground	Cube	Tree
Ground	G/G	G/C	G/T
Cube	C/G	C/C	C/T
Tree	T/G	T/C	T/T

Predicted Class

Obstacles
- Cube (C)
- Tree (T)

Non-Obstacle
- Ground (G)

Figure 6.6 Formulating the metrics for this study

Below is a summary of main steps of the proposed tree vs. solid obstacle identification method:

1- Set the *Cell_Size* to 50 cm and extract the features (predictors and response variable). This criterion is in line with the one in chapter 4 as both studies have similar natures.

2- Consolidate data for all time frames in one table to be used in the machine learning process.

3- Eliminate records with $Distance > 30m$. Similar to chapter 5, this limitation was set to reduce the data size, aiming to increase the processing speed while keeping essential data needed for the autonomous system to make reliable navigation decisions with respect to its immediate surroundings.

4- Locate those cells that contain solid obstacles (cubes) and label them as 1 which represents ‘cube’.

5- Locate those cells that contain trees and label them as 2 which represents ‘tree’. (Approximate the area covered by each tree using the area of the circle inscribed in the bounding box of each tree).

6- Label the remaining cell as 0 representing ‘ground’ cells.

7- Try all thirteen classification models to train the data for Scene 1. Use twelve extracted features i.e. *N, N_BH, Neighbor_Mean[N],* $\mu_{I,}$ σ_I^2, $\mu_I\_UBH,$ *Neighbor_Mean[Z],* $\mu_{Z,}$ σ_Z^2, $Min\ Z, Max\ Z$ and $Distance$ as predictors, and use *Label* as response variable (use 5-fold CV method).

8- Compare the performance of the classification algorithms and choose the best one (use ‘Obstacle Sensitivity’ and ‘Tree vs. Cube Precision’ as the main criteria).

9- Use the data from the first two scenes to train the selected model from step 8; Use the data from the third scene to test its performance.

10- If the resulting ‘Tree vs. Cube Precision’ from the previous step is greater than 90%, then go to step 11. Otherwise, go to step 15.

11- Use the data from Scene 1 and Scene 3 to train the selected model from step 8; Use the data from Scene 2 to test its performance.

12- If the resulting ‘Tree vs. Cube Precision’ from step 11 is greater than 90%, then go to step 13. Otherwise, go to step 15.

13- Use the data from Scene 2 and Scene 3 to train the selected model from step 8; Use the data from Scene 1 to test its performance.

14- If the resulting ‘Tree vs. Cube Precision’ from step 13 is greater than 90%, then model acceptance criterion is met, and the desired model is achieved. Otherwise, go to step 15.

15- Use model tuning techniques such as adding cost function; Alternatively add/remove features to increase model's performance with regard to the main acceptance criterion.

16- Re-train the new model with the data from the first two scenes. Use the data from the third scene to test its performance. Then, go to 10.

Aforementioned steps can be summarized in the form of a flow chart diagram. Figure 6.7 provides such an illustration.

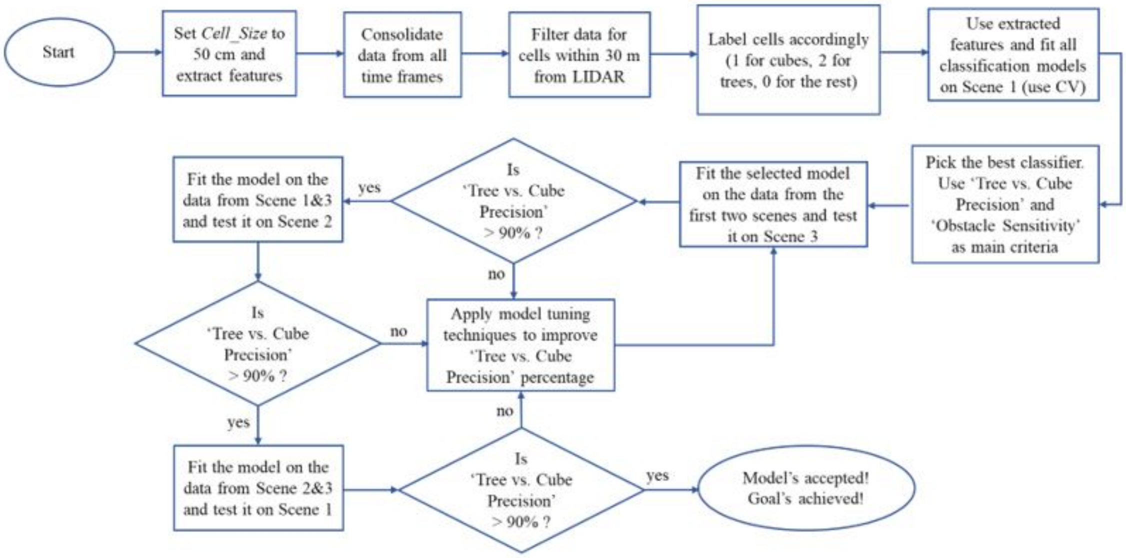

Start
Set *Cell_Size* to 50 cm and extract features
Consolidate data from all time frames
Filter data for cells within 30 m from LIDAR
Label cells accordingly (1 for cubes, 2 for trees, 0 for the rest)
Use extracted features and fit all classification models on Scene 1 (use CV)
Pick the best classifier. Use 'Tree vs. Cube Precision' and 'Obstacle Sensitivity' as main criteria
Fit the selected model on the data from the first two scenes and test it on Scene 3
Is 'Tree vs. Cube Precision' > 90% ?
yes
no
Fit the model on the data from Scene 1&3 and test it on Scene 2
Is 'Tree vs. Cube Precision' > 90% ?
no
yes
Apply model tuning techniques to improve 'Tree vs. Cube Precision' percentage
no
Fit the model on the data from Scene 2&3 and test it on Scene 1
Is 'Tree vs. Cube Precision' > 90% ?
yes
Model's accepted! Goal's achieved!

Figure 6.7 Flow chart diagram of the proposed methodology for the study

From the proposed steps, it can be interpreted that a model should meet two criteria in different steps to be accepted:

1- In the initial model selection step, two metrics had the higher importance to assess the performance of each model: 'Obstacle Sensitivity' and 'Tree vs. Cube Precision'. The former addresses the safe navigation concerns. It is specifically useful when the intent is to optimize for capturing all the obstacles. However, that comes at the cost of increased false positives i.e. cells that do not contain any solid obstacle but are classified otherwise. The latter addresses optimized path planning concerns. It is specifically useful when the intent is to expand the navigable area by excluding those cells that contain parts of trees that are deemed to be safe from the navigability standpoint. In this step, there is no predefined acceptance percentage value. The goal is to consider all the results and choose the model that have the overall best performance with regard to both of the aforementioned criteria.

2- After the initial model is selected, the rest of the efforts are focused on achieving a high 'Tree vs. Cube Precision' percentage (the rest of metrics have secondary importance). First, this criterion is assessed on the 5-fold CV results of the model fitted on the training data (data from Scene 1and Scene 2). Then, it was assessed on the test scene (Scene 3) using the trained model. In this context, a 'Tree vs. Cube Precision' percentage of higher than 90% is desired.

6.3 Results and Discussion

6.3.1 Visualization results based on cell labels

Visual investigation of the data can provide better insight about the model and how it can possibly be improved. Therefore, part of the discussion in this section considers the visual results from the study. As an example, after the scenes were gridded into x-y plain cells, each cell was labeled based on the type of object it contained. Three classes were defined: cells that contain a tree or a part of a tree, cells that contain a cube or a part of a cube and cells that do not contain tree or cube, but they might contain understory vegetation. The latter is considered as ground cells. Figure 6.8 through 6.10 illustrate the labeled scenes where trees were modeled as circles on the x-y plains. It is worth noting that in the labeling algorithm that was developed for this study, a cube has priority over a tree, meaning that if a cell contained both a partial/full tree and a partial/full cube, it was labeled as cube. It is because in the proposed methodology, safety precedes optimized path. Misclassified cells that label a cube as a tree would run the risk of severe collusion while the opposite might not be always the case.

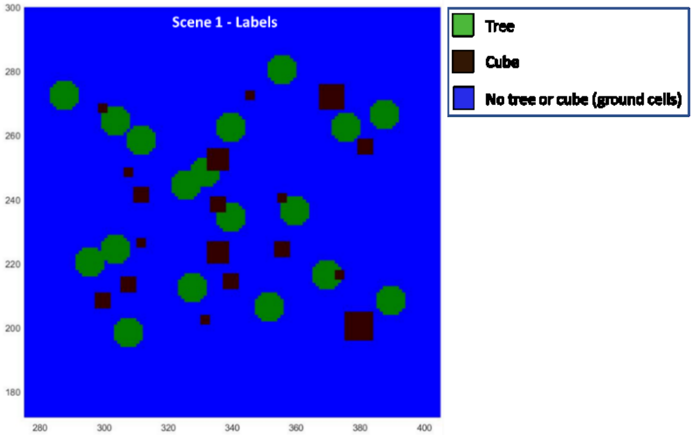


Figure 6.8 Labeled Scene 1

The overlap of cube and tree (brown spots that are partly or entirely surrounded by a green area) represent cells that contain both a tree and a cube but due to cube's priority over tree, the area is shown as brown (cube).

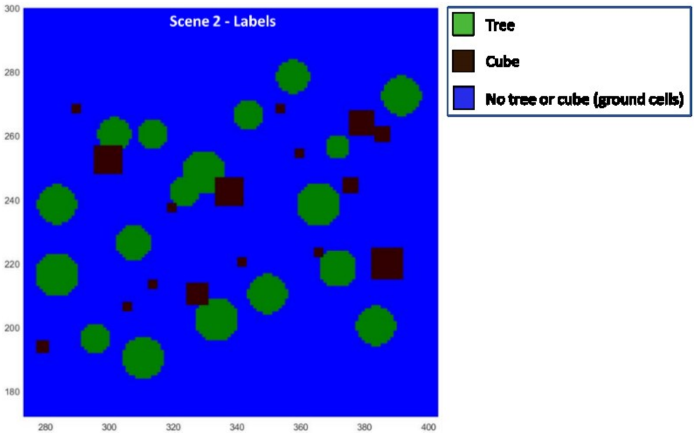


Figure 6.9 Labeled Scene 2

There are fewer cubes in this scene in Scene 1. Location and orientation of the trees are completely different than those in Scene 1. Also, cubes have different location and sizes.

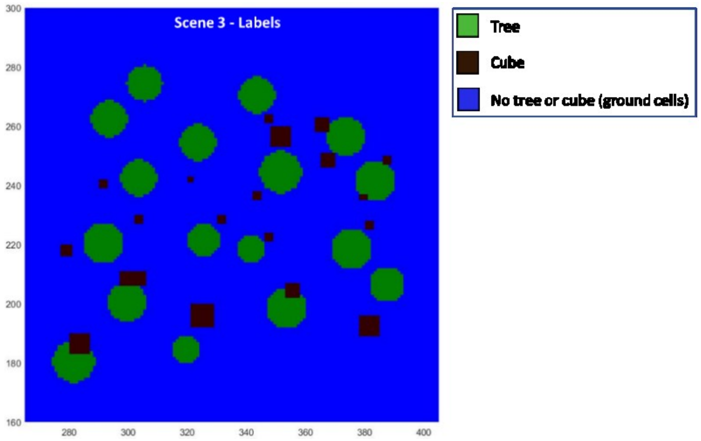


Figure 6.10 Labeled Scene 3

There are fewer trees but more cubes in this scene than in Scene 1 and Scene 2. To randomize the scenes, trees in this scene have different location and orientation than those in Scene 1 and Scene 2. Also, cubes have different location and sizes.

6.3.2 5-fold CV classification results for initial model selection

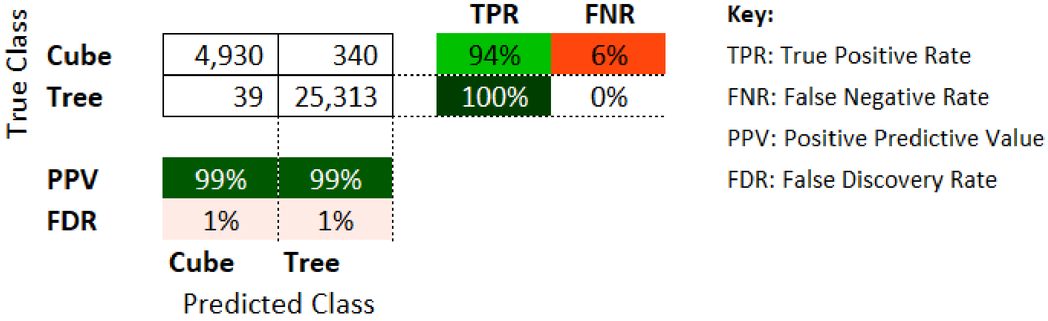
Table 6.2 summarizes the results for all of the thirteen classification models that were applied to scene 1 (step 7 of the proposed the methodology). The 'Bagged Trees' model showed an overall better performance than other models. It was the only model that consistently showed high results with regard to each metric. More specifically, it solely had the best performance with regard to four (out of six) metrics: Overall Accuracy, Obstacle Sensitivity, Obstacle Precision, and Tree vs. All Precision. It also showed a tied highest performance with regard to Tree vs. Cube Precision metric. Finally, it resulted in second best performance with regard to Cube vs. Tree Precision metric. Thus, Bagged Trees model was selected to proceed with in the next steps of this study.

Table 6.2 Results of classification models on Scene 1

Model	**Reference/Prediction**									**Obstacle Sensitivity**	**Obstacle Precision**	**Tree vs. All Precision**	**Tree vs. Cube Precision**	**Cube vs. Tree Precision**	**Overall Accuracy**
	G/G	G/ C	G/T	C/G	C/C	C/T	T/G	T/C	T/T						
Decision Trees (simple)	206,663	628	4,399	3,803	2,152	525	4,635	94	25,259	77%	85%	84%	98%	96%	94.3%
Decision Trees (medium)	207,338	172	4,180	3,109	2,849	522	4,659	49	25,280	79%	87%	84%	98%	98%	94.9%
Decision Trees (complex)	207,944	195	3,551	2,355	3,665	460	4,774	81	25,133	80%	89%	86%	98%	98%	95.4%
LDA	205,675	2,600	3,415	3,283	2,789	408	6,160	735	23,093	74%	82%	86%	98%	79%	93.3%
QDA	202,773	4,008	4,909	2,180	3,793	507	3,862	654	25,472	83%	77%	82%	98%	85%	93.5%
Ensemble (Subspace DA)	207,586	892	3,212	4,803	1,269	408	6,633	250	23,105	69%	86%	86%	98%	84%	93.5%
Ensemble (Boosted Trees)	207,313	82	4,295	2,483	3,520	477	4,566	25	25,397	81%	87%	84%	98%	99%	95.2%
Ensemble (Bagged Trees)	209,156	60	2,474	1,210	4,930	340	4,636	39	25,313	84%	92%	90%	99%	99%	96.5%
Ensemble (Subspace KNN)	208,615	48	3,027	2,995	3,038	447	5,888	12	24,088	76%	90%	87%	98%	100%	95.0%
KNN (Fine)	206,451	779	4,460	1,207	5,022	251	5,891	296	23,801	81%	85%	83%	99%	94%	94.8%
KNN (Medium)	208,838	129	2,723	1,523	4,660	297	5,574	105	24,309	81%	91%	89%	99%	98%	95.8%
KNN (Coarse)	208,242	185	3,263	2,020	4,057	403	5,420	97	24,471	80%	89%	87%	98%	98%	95.4%
KNN (Cosine)	208,201	139	3,350	1,437	4,727	316	5,166	102	24,720	82%	90%	87%	99%	98%	95.8%

Key
G: Ground
C: Cube
T: Tree

Figure 6.11 provides a closer look at the performance of the Bagged Trees algorithm with regard to tree vs. cube on the first scene. Positive Predictive Value of 99% which is also captured as ‘Tree vs. Cube Precision’ in Table 2 is the main metric that was considered in the next model selection steps.



True Class

True Class	Cube	Tree	TPR	FNR
Cube	4,930	340	94%	6%
Tree	39	25,313	100%	0%
PPV	99%	99%		
FDR	1%	1%		

Predicted Class

Key:
TPR: True Positive Rate
FNR: False Negative Rate
PPV: Positive Predictive Value
FDR: False Discovery Rate

Figure 6.11 Cube vs. Tree Focused Confusion Matrix for Scene 1

6.3.3 5-fold CV classification results for the final fitted models

According to the proposed methodology, to test the model on the new scenes on each round, one scene was removed from the training data and was later used to test the performance of the trained model. Each time, 5-fold cross validation technique was used to fit the model on the training data. Table 6.3 summarizes the 5-fold cross validation results for all three fitted models according to which, the 5-fold cross validation results for the main measure i.e. ‘Tree vs. Cube Precision’ were higher than 99% for all the three training datasets. The results indicate that the extracted features were able to produce promising accuracy values on the training data. Therefore, without further model enhancement, the model was ready be evaluated on the test scenes.

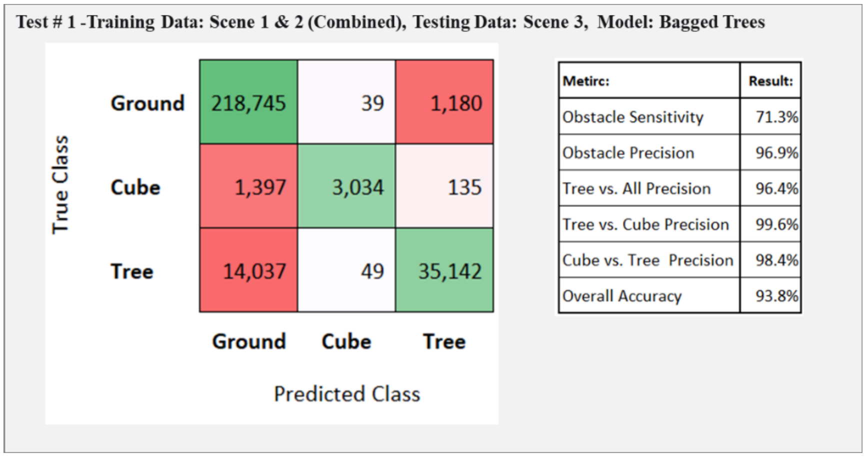
Table 6.3 Result of Bagged Trees model on training dataset

Training Data	Testing Data	Reference/Prediction									Obstacle Sensitivity	Obstacle Precision	Tree vs. All Precision	Tree vs. Cube Precision	Cube vs. Tree Precision	Overall Accuracy
		G/G	G/C	G/T	C/G	C/C	C/T	T/G	T/C	T/T						
Scene 1&2 (Combined)	Scene 3	407,696	153	7,908	1,663	12,369	541	14,186	93	61,232	82.4%	90.2%	87.9%	99.1%	99.3%	95.1%
Scene 1&3 (Combined)	Scene 2	423,907	137	7,610	2,331	8,162	553	14,570	81	64,565	81.3%	90.4%	88.8%	99.2%	99.0%	95.2%
Scene 2&3 (Combined)	Scene 1	417,716	131	6,184	1,697	10,551	411	16,849	106	77,796	82.7%	93.4%	92.2%	99.5%	99.0%	95.2%

All the Tree vs. Cube Precision values are higher than 99%

6.3.4 Prediction results on the test scenes

After the selected model was fitted to each training dataset, it was tested against the corresponding testing scene. Figure 6.12 through 6.14 provide the confusion matrix and other prediction results for all three test scenes. According to the results, Tree vs. Cube Precision value for each test scene was well above the passing criterion for this study. Thus, the results confirm that the extracted features and the final model were able to produce accurate enough results meeting the model acceptance criterion (i.e. Tree vs. Cube Precision > 90%). However, it can be noted that 'Obstacle Sensitivity' is always lower than the main evaluation metric (i.e. Tree vs. Cube Precision). The reason is that the main focus of this study was on developing the capability to precisely differentiate trees from the cubes which is well addressed by the metric 'Tree vs. Cube Precision'. Therefore, the model was optimized according to that metric. Had the model in this study been optimized to produce higher 'Obstacle Sensitivity' results, the 'Tree vs. Cube Precision' would have been adversely affected because a model can have a high sensitivity by recalling most (if not all) of the obstacles as a result of classifying a lot of non-obstacle objects as obstacles. However, such a model would have lower 'Precision' values for the exact same reason (i.e. a lot of misclassified objects as obstacles). Therefore, in real applications, autonomous systems should utilize different machine learning algorithms simultaneously, each one developed to address a specific concern.

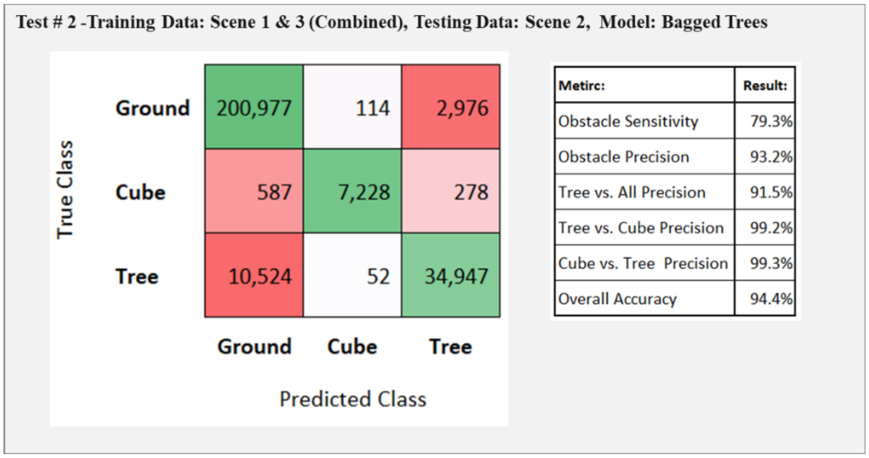


Test # 1 -Training Data: Scene 1 & 2 (Combined), Testing Data: Scene 3, Model: Bagged Trees

True Class	Ground	Cube	Tree
Ground	218,745	39	1,180
Cube	1,397	3,034	135
Tree	14,037	49	35,142
	Predicted Class		

Metirc:	Result:
Obstacle Sensitivity	71.3%
Obstacle Precision	96.9%
Tree vs. All Precision	96.4%
Tree vs. Cube Precision	99.6%
Cube vs. Tree Precision	98.4%
Overall Accuracy	93.8%

Figure 6.12 Result of Bagged Trees model on the test scene # 1 (Scene 3)

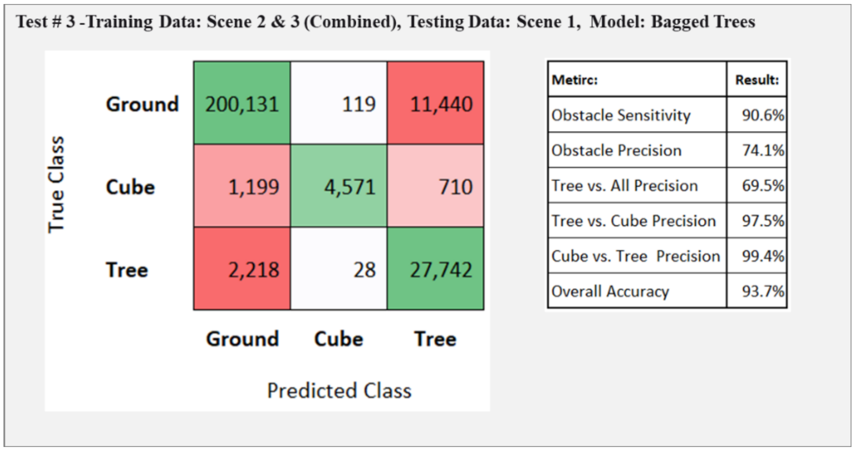


Test # 2 -Training Data: Scene 1 & 3 (Combined), Testing Data: Scene 2, Model: Bagged Trees

True Class	Ground	Cube	Tree
Ground	200,977	114	2,976
Cube	587	7,228	278
Tree	10,524	52	34,947
	Predicted Class		

Metirc:	Result:
Obstacle Sensitivity	79.3%
Obstacle Precision	93.2%
Tree vs. All Precision	91.5%
Tree vs. Cube Precision	99.2%
Cube vs. Tree Precision	99.3%
Overall Accuracy	94.4%

Figure 6.13 Result of Bagged Trees model on the test scene # 2 (Scene 2)



Test # 3 -Training Data: Scene 2 & 3 (Combined), Testing Data: Scene 1, Model: Bagged Trees

True Class	Ground	Cube	Tree
Ground	200,131	119	11,440
Cube	1,199	4,571	710
Tree	2,218	28	27,742
	Predicted Class		

Metirc:	Result:
Obstacle Sensitivity	90.6%
Obstacle Precision	74.1%
Tree vs. All Precision	69.5%
Tree vs. Cube Precision	97.5%
Cube vs. Tree Precision	99.4%
Overall Accuracy	93.7%

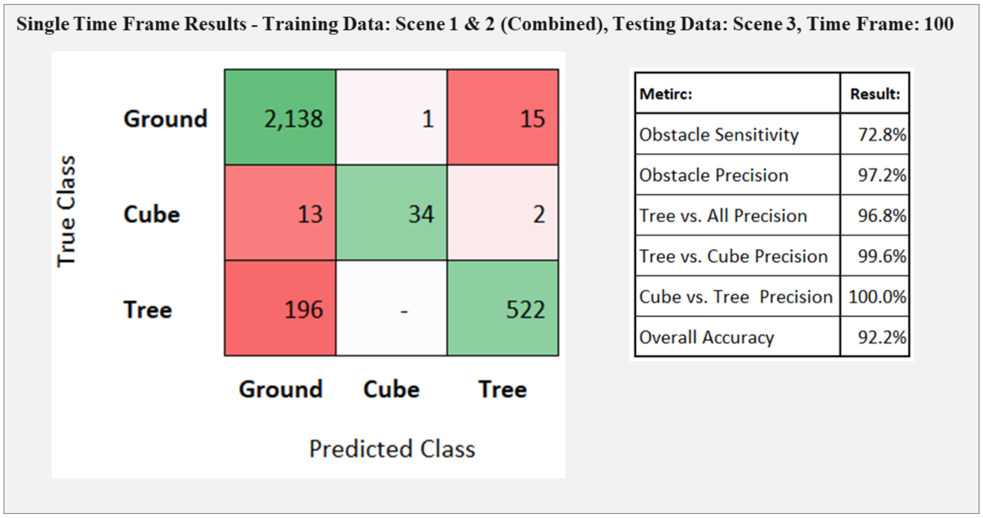
Figure 6.14 Result of Bagged Trees model on the test scene # 3 (Scene 1)

Table 6.4 presents the average performance of the final model on the three test scenes. With regard to the most important metric i.e. Tree vs. Cube Precision, the average value of 98.8% was well above the minimum acceptance criterion.

Table 6.4 Overall Prediction Results (All Test Scenes)

	Metric					
	Obstacle Sensitivity	**Obstacle Precision**	**Tree vs. All Precision**	**Tree vs. Cube Precision**	**Cube vs. Tree Precision**	**Overall Accuracy**
Avg. Performance: **(All Test Scenes)**	80.4%	88.1%	85.8%	98.8%	99.0%	94.0%

To better investigate and understand the performance of the selected model on the test scenes, the trained model was used to make prediction on one single timeframe in the first test scene (Scene 3). Time frame # 100 was selected as it represents the time that the vehicle was approximately in the middle of the scene. Figure 6.15 provides the confusion matrix and other prediction results for the time frame # 100.



Single Time Frame Results - Training Data: Scene 1 & 2 (Combined), Testing Data: Scene 3, Time Frame: 100

True Class	Ground	Cube	Tree
Ground	2,138	1	15
Cube	13	34	2
Tree	196	-	522
	Predicted Class		

Metirc:	Result:
Obstacle Sensitivity	72.8%
Obstacle Precision	97.2%
Tree vs. All Precision	96.8%
Tree vs. Cube Precision	99.6%
Cube vs. Tree Precision	100.0%
Overall Accuracy	92.2%

Figure 6.15 Result of Bagged Trees model on data frame 100 from the 1st test scene (Scene 3) The precision of 99.6% is in line with the second model selection criterion.

Figure 6.16 provides a visual representation of the true labels (the image on the left) and the predicted labels (the image on the right) for the time frame # 100 of the first test scene. From the visual standpoint, the algorithms did a good job classifying the scene. From the images, however, it can be realized that there was a lot of cells (colored in grey) for which there was no data captured on that specific time frame. That is justifiable as LIDAR's sight gets blocked by the obstacles around it. It is specifically visible in the areas behind the three trees that are closer to the LIDAR in the middle of the scene. Grey circular sectors behind the green spots represent the cells that did not reflect any LIDAR beam due to the occlusion caused by the trees. Thus, the predictive model did not have any input to make prediction of the respective labels.

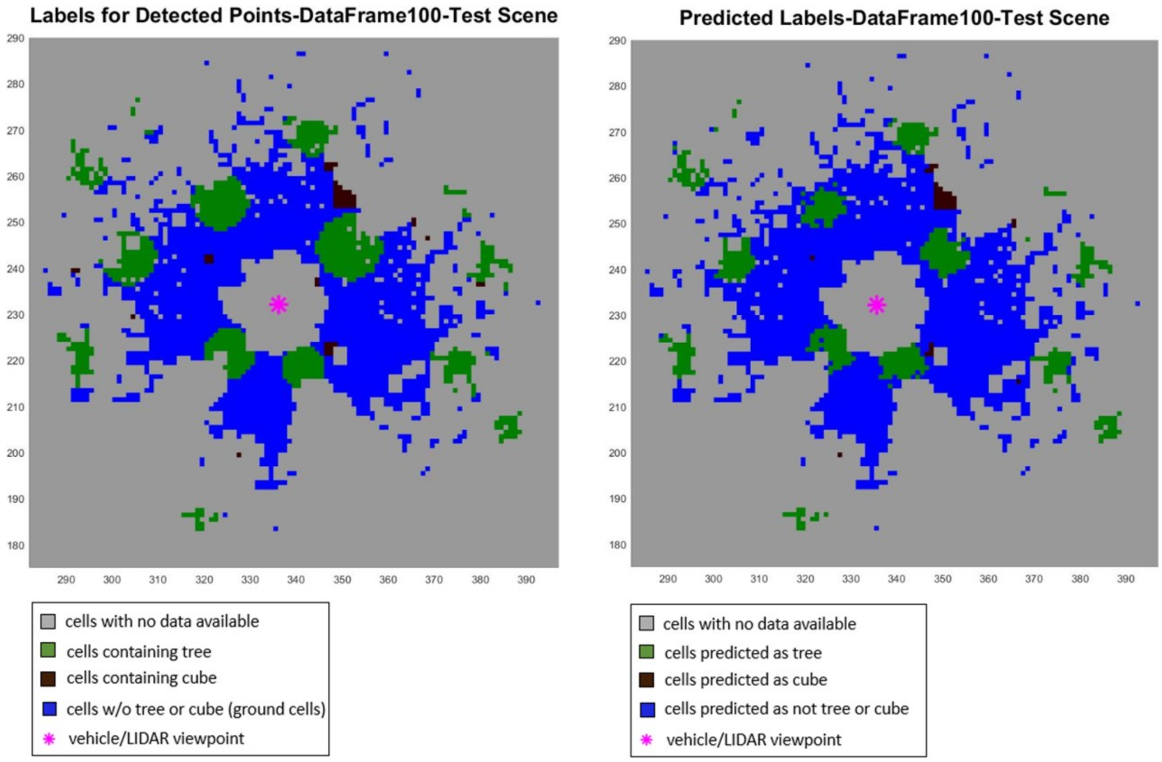


Figure 6.16 Comparison of True Labels and Predicted Labels for Single Timeframe (Scene 3)

Figure 6.17 is another visual representation of the algorithm's prediction performance on the time frame # 100 of the first test scene. It essentially merges the information from the two images in Figure 6.16 and represents an all-in-one image. In the context of tree vs. cube predictions, one of the main misclassifications that is particularly important to avoid is the cells containing cubes that were wrongly classified as tree (depicted in red in Figure 6.17). As it was discussed earlier in this chapter, this type of misclassification can pose safety risks. This visual representation confirms a robust performance of the model with regard to that type of misclassification as there are only two red spots representing two cells that were incorrectly classified by the model as tree. Those red spots are also far enough from the LIDAR that would technically eliminate any immediate risk of collision. It is possible that as the vehicle gets closer to those spots, the predictive algorithm corrects its initial misclassification. Spots depicted in cyan color represents cells that contain tree but were incorrectly classified as cube. While these misclassified instances were more closer to the vehicle's position, they would not pose safety risks. Rather, they might result in suboptimal path solutions which is of secondary importance in this context. Yellow spots in the figure represent other type of misclassified cells such as ground cells classified as tree.

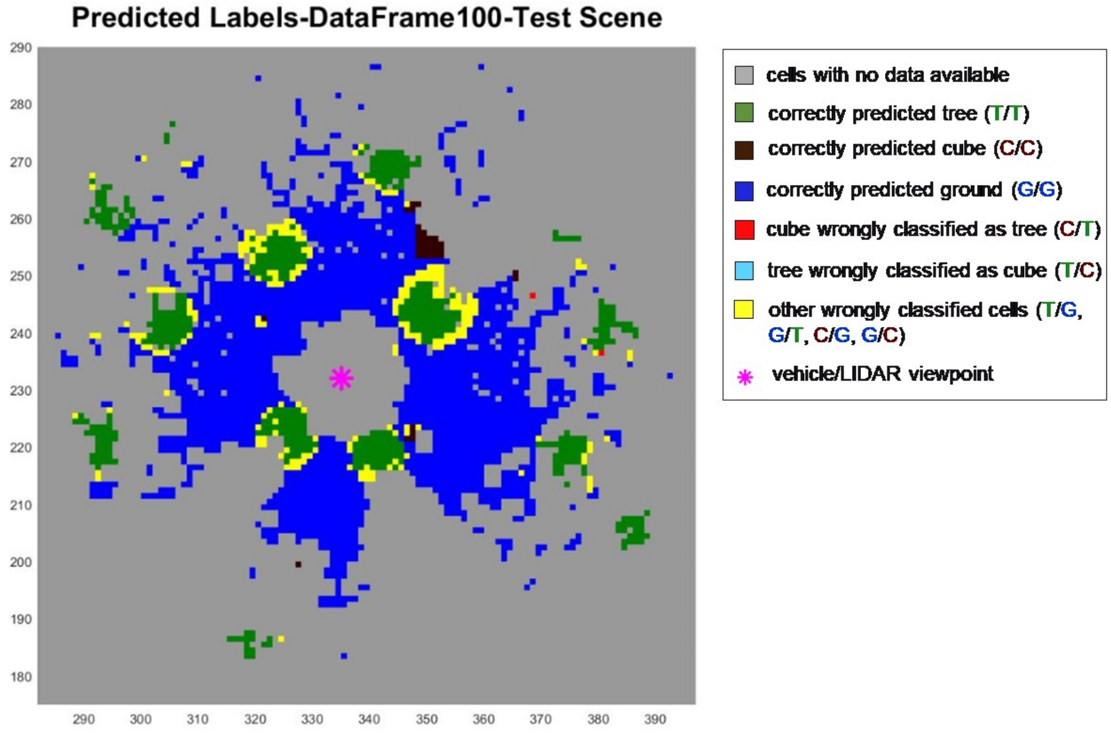


Figure 6.17 Visual Prediction Performance of the Model for Data Frame 100 (Scene 3)

6.4 Conclusion

In this chapter, a framework was proposed to enable AGVs identify trees in an obstacle-rich forested environment. Fourteen features were extracted out of which three were new features

(*N_BH,* μ_{I_UBH} and *Neighbor_Mean[Z]*) that were specifically meant to address the problem under study in this chapter i.e. identification of trees in an obstacle-rich forested environment for off-road autonomous ground vehicles. Thirteen classification models were tested. Six metrics were used to select the initial model. Two out of those six metrics, i.e. model's sensitivity in identifying obstacles (including trees and solid cubes) and model's precision in distinguishing trees from solid cubes, had the higher importance in identifying the initial model. The results showed an overall superior performance in the model that uses Bootstrap Aggregation (also known as Bagged Trees technique) which is an ensemble method. This is an ensemble method that combines the predictions from multiple machine learning algorithms together to make more accurate predictions than any individual model (Brownlee 2019). Bootstrap Aggregation is a useful technique when the goal is to reduce the variance of decision trees within an ensemble method (Nagpal 2017).

Using 5-fold cross validation, the selected model was fitted to three training datasets. Subsequently, each fitted model was used to make predictions on its corresponding test scene. The results met the criterion that was set for this study i.e. the resulting 'Tree vs. Obstacle Precision' values for each test scene was higher than 90%. The numbers confirmed that the proposed methodology including the extracted features and selected parameters were relevant enough to result in the desired Tree vs. Obstacle Precision level.

For future work, one of the opportunities would be the identification of cells that contain foliage and parts of the trees that do not pose safety risk to the AGV. In this context, the future study should focus on classifying different parts of a tree rather than regarding the entire tree as an obstacle. Similar to the previous studies of this dissertation, the scenes are set to be flat. Thus, another future research opportunity exists in considering scenes with steep slopes or hilly terrains that are covered with trees and contain solid obstacles. Additionally, to simplify the labeling in this study, trees were modeled as complete circles on the x-y plane. Such an approximation resulted in many cells with the true label of 'Tree' to be classified as 'Ground' by the algorithm. It is very important to mention that while such misclassification adversely affect the accuracy of the model, it does not mean the 'predicted' labels were incorrect. From the visual illustration of the predictions, it can be interpreted that in many of those instances, the algorithm was able to correctly identify the real location of the cells containing the trees. However, the 'true' labels were based on the approximate method which -in many cases- classified cells that did not contain part of a tree as 'Tree'. An improvement opportunity to the current work would be the use of more accurate methods for labeling the cells that contain trees. Finally, cube-shaped objects were used in this study as the representation of solid obstacles. This enabled a simpler and more accurate labeling of solid obstacles. An extension to this work could be the use of irregular-shaped solid obstacles in the scenes that would more represent the real objects commonly found in the context of off-road environments.

CHAPTER VII

CONTRIBUTION

This dissertation integrates past studies related to traversability analysis for AGVs in unstructured forested environments and proposes novel approaches that are expected to offer higher performance compared to previously developed models. The term ‘higher performance’ can be considered as a combination of different metrics such as model accuracy, speed, and capability to perform the computation on-line. For each topic, new features were introduced and extracted that were used in the proposed predictive algorithms. Unique contributions of each work are listed separately in the following subsections.

7.1 Assessing impact of understory vegetation density on solid obstacle detection for off-road AGVs

Understory vegetation is an inherent element of unstructured forested areas. Therefore, the off-road applications of autonomous systems should acknowledge its existence and consider its effects on the accuracy of their traversability analyses. In contrast to previous studies that do not provide quantitative details about the distribution of vegetation, the proposed framework defined and quantified density levels and studied how the obstacle detection accuracy was affected by the understory vegetation density by leveraging machine learning methods. It proposed novel cumulative cell-wise features extracted for obstacle detection for off-road AGVs. The proposed method for quantifying the density levels is scalable and can be used for interpolation of detectability in other density levels. The results of this work showed that the increase in the density of understory vegetation adversely affected the classification performance in correctly detecting solid obstacles. This finding highlighted the significant impact of understory density on safe traversability, which in turn resulted in an extended study as a separate work that was presented in CHAPTER V of this dissertation. The findings, however, are limited to the use of one type of understory vegetation plant i.e. couch grass. Use of different vegetation types (separately or combined) could result in a broader understanding of the understory vegetation density's impact on solid obstacle detectability of AGVs. This work was accepted and published by *ASME Letters in Dynamic Systems and Control* in July 2020.

7.2 Traversability risk mapping through understory vegetation with density estimation for off-road AGVs

CHAPTER IV confirmed the significant impact of understory vegetation density on solid obstacle detectability of autonomous systems. The results highlighted the need for quantifying the understory density for safe traversability purposes. Such a topic, however, was not studied previously in the context of off-road autonomous systems. There is, however, some prior work in the context of canopy estimation which studies the forest canopy from above the forested area (and not from the vehicle height level). The study in CHAPTER V was aimed to fill such a gap in the context of off-road autonomy. It introduced new features for estimating the understory vegetation density. Moreover, it proposed a new framework for developing traversability risk maps based on understory density estimation. Additionally, work 2 studied the impact of *Cell_Size* in a 2D gridded plain on the accuracy of understory density estimating models. The results of the density estimation algorithm confirmed the findings of CHAPTER IV where increased understory density resulted in the lower sensitivity values. Also, from the study, it was concluded that an increase in *Cell_Size* improved adjusted R-squared values. Given that the larger values for *Cell_Size* imply more aggregation of scene data, the finding of this study signifies the importance of optimizing predictive models based on a balance between adjusted R-squared (as an indication of the model accuracy) and *Cell_Size* (as an indication of visibility resolution). Furthermore, practical guidelines were provided in choosing the right *Cell_Size* given different traversability accuracy requirements. The predictive model, however, was trained and tested based on one understory vegetation type (i.e. couch grass). To better evaluate the performance of the proposed model, however, it is of interest to test the model on different understory vegetation types. This work was submitted to *Journal of Autonomous Vehicles and Systems* in July 2020.

7.3 Identification of trees in an obstacle-rich forested environment for off-road AGVs

In the context of forested environments, the ability of the scene perception algorithm to correctly distinguish trees from solid obstacles is very important because misclassification of trees as solid obstacles would unnecessarily limit the traversable area. The entire idea of differentiation between trees and solid obstacles for optimized path planning is a new topic that was not covered in the prior studies. Furthermore, different machine learning methods result in different accuracy values. They also might have significantly different run-times. All these factors can be highly impactful in actual autonomous applications. However, these factors have been mostly ignored or disregarded in the prior work. To address those research gaps, the study in CHAPTER VI introduced new features and proposed a classification-based machine learning framework that produced very high accuracy and precision results. Using LIDAR-only approach (without data fusion) to classify forested terrain objects with such a high precision (as in the proposed work in this dissertation) was not found in the prior work. The results confirmed that the proposed methodology including the extracted features and selected parameters were relevant enough to result in the desired prediction accuracy level to distinguish between obstacle and non-obstacle regions, as well as tree and solid obstacle regions. Besides, CHAPTER VI studied a wide range of statistically interpretable machine learning methods and found the one which resulted in the best performance given the format and nature of the dataset. However, it is worth mentioning that the results were based on the existence of one type of tree in the scenes (oak tree). The robustness of the proposed model in the environments with different types of trees was not thoroughly studied.

CHAPTER VIII

FUTURE OPPORTUNITIES

Future opportunities exist in addressing some of the limitations of this research and by relaxing some of the assumptions of this dissertation. For instance, incorporating the following topics to the current research are expected to enhance it and can be considered as future research opportunities:

8.1 Severe Weather Conditions

The simulation of data for this dissertation assumed a clear weather condition. However, the degradation in LIDAR performance due to the rain or severe weather conditions is a known fact and has been even quantified (Goodin et al. 2019). Future research opportunity exists in considering the effect of severe weather conditions in solid obstacle detection capability of the proposed algorithms in this dissertation. Also, accommodating weather variations can add another dimension to the proposed traversability risk maps framework that currently only considers the density of the understory for quantifying the risk.

8.2 Hilly Terrains

In this dissertation, all the developed scenes (including training and testing scenes) were designed to be completely flat. When developing models for classifying different objects within a scene, it is important to test the performance of the model on sloped trains. It is because the correct identification of objects is more difficult where slopes exist (McDaniel et al. 2012). An extension to the work that was presented in this study, could be testing and enhancing the accuracy of the proposed algorithms where terrains are not completely flat. The extreme case in this context can be considered as hilly terrains where both positive and negative slopes exist within the same scene.

8.3 Solid Obstacle Characterization in Scene Development and Validation

To model solid obstacles, cube-shaped objects were used in this dissertation. That is because estimating the area a cube-shaped object occupies is straight forward and simple. Those estimates were used in labeling the cells containing solid obstacles. Apparently, use of rocks or other natural solid obstacles in the scenes would make it much more challenging to have the cells within a scene reliably labeled. Also, for the simplicity of calculations in this dissertation, circular shapes were used to model the area a tree crown covers on the x-y plain. While circles might very well represent tree crowns on a 2D basis, they cannot provide precise information about the area that the tree crown covers. A solution that can be applied to both of the aforementioned cases i.e. solid obstacles with irregular shapes and tree crowns, is the use of convex hull technique. Specifically, using convex hull approach for estimating the area a tree crown covers on the x-y plain is deemed to improve the accuracy of the model that was proposed in the third work topic of this dissertation.

8.4 Dynamic Environments

All the studies in this dissertation were implemented under the assumption that the only moving object in the scenes is the vehicle. Such an assumption made it significantly easier to develop the proposed algorithms. That is because the data from previous time frames were valid throughout the navigation. However, in real applications, the possible presence of other moving objects such as vehicles, people, and animals add a new layer of difficulty to the path planning algorithms as they cannot simply rely on the scene modeling based on the previous time frames. A future research opportunity exists in modeling dynamic environments in which the course of movement for each object should be modeled and assessed so that the control algorithm can reliably react to them.

REFERENCES

Ahtiainen, Juhana, Todor Stoyanov, and Jari Saarinen. 2017. "Normal Distributions Transform Traversability Maps: LIDAR-Only Approach for Traversability Mapping in Outdoor Envoronments." *Journal of Field Robotics* 34(3): 600–621.

Arnold, Eduardo et al. 2019. "A Survey on 3D Object Detection Methods for Autonomous Driving Applications." *IEEE Transactions on Intelligent Transportation Systems* 20(10): 3782–95.

Bajracharya, Max et al. 2009. "Autonomous Off-Road Navigation with End-to-End Learning for the LAGR Program." *Journal of Field Robotics* 26(1): 3–25.

Balakirsky, Stephan, C Scrapper, S Carpin, and M Lewis. 2006. "USARSim: Providing a Framework for Multi-Robot Performance Evaluation." In *Proc. of the Performance Metrics for Intelligent Systems (PerMIS) Workshop*, , 98–102.

Bares, John et al. 1989. "Ambler: An Autonomous Rover for Planetary Exploration." *Computer* 22(6): 18–26.

Biber, Peter, and Wolfgang Straßer. 2003. "The Normal Distributions Transform: A New Approach to Laser Scan Matching." In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, , 2743–48.

Bradley, David M., Scott M. Thayer, Anthony Stentz, and Peter Rander. 2004. "Vegetation Detection for Mobile Robot Navigation." *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-04-12*.

Brownlee, Jason. 2019. "Bagging and Random Forest Ensemble Algorithms for Machine Learning." www.machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/ (January 6, 2020).

Buehler, Martin, Karl Iagnemma, and Sanjiv Singh. 2007. 36 *The 2005 DARPA Grand Challenge: The Great Robot Race*. Springer.

Campbell, Michael J et al. 2018. "Quantifying Understory Vegetation Density Using Small-Footprint Airborne Lidar." *Remote sensing of environment* 215: 330–42.

Carpin, Stefano et al. 2007. "USARSim: A Robot Simulator for Research and Education." In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, , 1400–1405.

Chavez-Garcia, R. Omar, Jerome Guzzi, Luca M. Gambardella, and Alessandro Giusti. 2018. "Learning Ground Traversability from Simulations." *IEEE Robotics and Automation Letters* 3(3): 1695–1702.

Choi, Jaewoong et al. 2012. "Environment-Detection-and-Mapping Algorithm for Autonomous Driving in Rural or Off-Road Environment." *IEEE Transactions on Intelligent Transportation Systems* 13(2): 974–82.

Clark, Matthew L., Dar A. Roberts, John J. Ewel, and David B. Clark. 2011a. "Estimation of Tropical Rain Forest Aboveground Biomass with Small-Footprint Lidar and Hyperspectral Sensors." *Remote Sensing of Environment* 115(11): 2931–42. http://dx.doi.org/10.1016/j.rse.2010.08.029.

Clark, Matthew L, Dar A Roberts, John J Ewel, and David B Clark. 2011b. "Estimation of Tropical Rain Forest Aboveground Biomass with Small-Footprint Lidar and Hyperspectral Sensors." *Remote Sensing of Environment* 115(11): 2931–42.

"Couch Grass." www.xfrog.com/product/GC08.html (June 2, 2020).

Dalponte, Michele, Lorenzo Bruzzone, and Damiano Gianelle. 2011. "A System for the Estimation of Single-Tree Stem Diameter and Volume Using Multireturn LiDAR Data." *IEEE Transactions on Geoscience and Remote Sensing* 49(7): 2479–90.

Drake, Jason B et al. 2002. "Estimation of Tropical Forest Structural Characteristics Using Large-Footprint Lidar." *Remote Sensing of Environment* 79(2–3): 305–19.

Durst, Phillip J et al. 2012. "A Real-Time, Interactive Simulation Environment for Unmanned Ground Vehicles: The Autonomous Navigation Virtual Environment Laboratory (ANVEL)." In *2012 Fifth International Conference on Information and Computing Science*, , 7–10.

Epshtein, Uriel, and Charles Faint. 2019. "That's Logistics: The Autonomous Future of the Army's Battlefield Supply Chain." http://mwi.usma.edu/thats-logistics-autonomous-future-armys-battlefield-supply-chain/.

Falkowski, Michael J et al. 2010. "Landscape-Scale Parameterization of a Tree-Level Forest Growth Model: A k-Nearest Neighbor Imputation Approach Incorporating LiDAR Data." *Canadian Journal of Forest Research* 40(2): 184–99.

Foroutan, Morteza, Christopher T. Goodin, and Wenmeng Tian. 2020. "Assessing Impact of Understory Vegetation Density on Solid Obstacle Detection for Off-Road Autonomous Ground Vehicles." *ASME Letters in Dynamic Systems and Control*: 1–22.

Franz, Gerald, Hanspeter Mallot, Jan Wiener, and Kognitive Neurowissenschaft. 2005. “Graph-Based Models of Space in Architecture and Cognitive Science-a Comparative Analysis.” In *Proceedings of the 17th International Conference on Systems Research, Informatics and Cybernetics*,.

Garrido, Santiago, Mar\'\ia Malfaz, and Dolores Blanco. 2013. “Application of the Fast Marching Method for Outdoor Motion Planning in Robotics.” *Robotics and Autonomous Systems* 61(2): 106–14.

General Motors. 2018. *2018 Self-Driving Safety Report 1*. www.gm.com/content/dam/company/docs/us/en/gmcom/gmsafetyreport.pdf.

Goodin, Christopher, Justin T Carrillo, and David P Mcinnis. 2017. “Unmanned Ground Vehicle Simulation with the Virtual Autonomous Navigation Environment.”

Goodin, Christopher, Daniel Carruth, Matthew Doude, and Christopher Hudson. 2019. “Predicting the Influence of Rain on LIDAR in ADAS.” *Electronics* 8(1): 89.

Hebert, Martial, Nicolas Vandapel, Stefan Keller, and R Rao Donamukkala. 2002. “Evaluation and Comparison of Terrain Classification Techniques from LADAR Data for Autonomous Navigation.” In *23rd Army Science Conference*,.

Heckman, Nicholas, Jean-François Lalonde, Nicolas Vandapel, and Martial Hebert. 2007. “Potential Negative Obstacle Detection by Occlusion Labeling.” In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, , 2168–73.

Hellström, Thomas, Pär Lärkeryd, Tomas Nordfjell, and Ola Ringdahl. 2009. “Autonomous Forest Vehicles: Historic, Envisioned, and State-of-the-Art.” *International Journal of Forest Engineering* 20(1): 31–38.

Hornung, Armin et al. 2013. “OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees.” *Autonomous robots* 34(3): 189–206.

Howard, Andrew et al. 2006. “Towards Learned Traversability for Robot Navigation: From Underfoot to the Far Field.” *Journal of Field Robotics* 23(11–12): 1005–17.

Hudak, Andrew T et al. 2006. “Regression Modeling and Mapping of Coniferous Forest Basal Area and Tree Density from Discrete-Return Lidar and Multispectral Satellite Data.” *Canadian Journal of Remote Sensing* 32(2): 126–38.

Hudson, Christopher R, Chris Goodin, Matthew Doude, and Daniel W Carruth. 2018. “Analysis of Dual Lidar Placement for Off-Road Autonomy Using Mavs.” In *2018 World Symposium on Digital Intelligence for Systems and Machines (DISA)*, , 137–42.

Jo, Kichun et al. 2014. “Development of Autonomous Car—Part I: Distributed System Architecture and Development Process.” *IEEE Transactions on Industrial Electronics* 61(12): 7131–40.

Koenig, Nathan, and Andrew Howard. 2004. “Design and Use Paradigms for Gazebo, an Open-Source Multi-Robot Simulator.” In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, , 2149–54.

Lacaze, Alberto, Karl Murphy, and Mark DelGiorno. 2002. “Autonomous Mobility for the Demo III Experimental Unmanned Vehicles.” In *In Assoc. for Unmanned Vehicle Systems Int. Conf. on Unmanned Vehicles (AUVSI 02*,.

Lalonde, Jean-François, Nicolas Vandapel, Daniel F Huber, and Martial Hebert. 2006. “Natural Terrain Classification Using Three-Dimensional Ladar Data for Ground Robot Mobility.” *Journal of field robotics* 23(10): 839–61.

Lang, Tobias, Christian Plagemann, and Wolfram Burgard. 2007. “Adaptive Non-Stationary Kernel Regression for Terrain Modeling.” In *Robotics: Science and Systems*,.

De Luca, Alessandro, Wayne Book, B Siciliano, and O Khatib. 2008. “Springer Handbook of Robotics.”

Macedo, Jose, Roberto Manduchi, and Larry Matthies. 2001. “Ladar-Based Discrimination of Grass from Obstacles for Autonomous Navigation.” In *Experimental Robotics VII*, Springer, 111–20.

Magnusson, Martin. 2009. “The Three-Dimensional Normal-Distributions Transform: An Efficient Representation for Registration, Surface Analysis, and Loop Detection.” Örebro universitet.

Manduchi, Roberto, Andres Castano, Ashit Talukder, and Larry Matthies. 2005. “Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation.” *Autonomous robots* 18(1): 81–102.

Martin, Steven, Liz Murphy, and Peter Corke. 2013. “Building Large Scale Traversability Maps Using Vehicle Experience.” In *Experimental Robotics*, , 891–905.

Martinuzzi, Sebastián et al. 2009. “Mapping Snags and Understory Shrubs for a LiDAR-Based Assessment of Wildlife Habitat Suitability.” *Remote Sensing of Environment* 113(12): 2533–46.

McDaniel, Matthew W et al. 2012. “Terrain Classification and Identification of Tree Stems Using Ground-Based LiDAR.” *Journal of Field Robotics* 29(6): 891–910.

Moravec, Hans, and Alberto Elfes. 1985. “High Resolution Maps from Wide Angle Sonar.” In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, , 116–21.

Myneni, Ranga B, Forrest G Hall, Piers J Sellers, and Alexander L Marshak. 1995. “The Interpretation of Spectral Vegetation Indexes.” *IEEE Transactions on Geoscience and Remote Sensing* 33(2): 481–86.

Nagpal, Anuja. 2017. “Decision Tree Ensembles- Bagging and Boosting.” www.towardsdatascience.com/decision-tree-ensembles-bagging-and-boosting-266a8ba60fd9 (January 6, 2020).

Nefian, Ara V, and Gary R Bradski. 2006. “Detection of Drivable Corridors for Off-Road Autonomous Navigation.” *2006 International Conference on Image Processing*: 3025–28.

Nguyen, D-V et al. 2012. “A Novel Approach for a Double-Check of Passable Vegetation Detection in Autonomous Ground Vehicles.” In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, , 230–36.

Pendleton, Scott et al. 2017. “Perception, Planning, Control, and Coordination for Autonomous Vehicles.” *Machines* 5(1): 6.

Reutebuch, Stephen E, Hans-Erik Andersen, and Robert J McGaughey. 2005. “Light Detection and Ranging (LIDAR): An Emerging Tool for Multiple Resource Inventory.” *Journal of Forestry* 103(6): 286–92.

Saarinen, Jari P, Henrik Andreasson, Todor Stoyanov, and Achim J Lilienthal. 2013. “3D Normal Distributions Transform Occupancy Maps: An Efficient Representation for Mapping in Dynamic Environments.” *The International Journal of Robotics Research* 32(14): 1627–44.

Santamaria-Navarro, Àngel, Ernesto H Teniente, Mart\'\i Morta, and Juan Andrade-Cetto. 2015. “Terrain Classification in Complex Three-Dimensional Outdoor Environments.” *Journal of Field Robotics* 32(1): 42–60.

Silva, Varuna De, Jamie Roche, Ahmet Kondoz, and Senior Member. 2017. “Fusion of LiDAR and Camera Sensor Data for Environment Sensing in Driverless Vehicles.” *Loughborough’s Research Repository*. http://repository.lboro.ac.uk/articles/Fusion_of_LiDAR_and_camera_sensor_data_for_environment_sensing_in_driverless_vehicles/9466361/1.

Stiller, Christoph et al. 2015. “Perception and Planning for Autonomous Vehicles [Guest Editorial].” *IEEE Intelligent Transportation Systems Magazine* 7(1): 6–7.

Stoyanov, Todor, Martin Magnusson, Henrik Andreasson, and Achim J Lilienthal. 2010. “Path Planning in 3D Environments Using the Normal Distributions Transform.” In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, , 3263–68.

Stoyanov, Todor, Rasoul Mojtahedzadeh, Henrik Andreasson, and Achim J Lilienthal. 2013. “Comparative Evaluation of Range Sensor Accuracy for Indoor Mobile Robotics and Automated Logistics Applications.” *Robotics and Autonomous Systems* 61(10): 1094–1105.

TESLA. 2020. *Tesla Vehicle Safety Report*. www.tesla.com/VehicleSafetyReport.

Vandapel, Nicolas, Daniel F D.F. Huber, Anuj Kapuria, and Martial Hebert. 2004. "Natural Terrain Classification Using 3-d Ladar Data." In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, , 5117–22. http://ieeexplore.ieee.org/document/1302529/.

Ward, Luke. 2019. "12 Facts About Oak Trees You Wood Not Believe." www.thefactsite.com/oak-tree-facts.

Waymo. 2018. "Waymo Safety Report: On the Road to Fully Self-Driving." *Energy and the Environment*. http://storage.googleapis.com/sdc-prod/v1/safety-report/Safety Report 2018.pdf.

Wellington, Carl, Aaron Courville, and Anthony Stentz. 2006. "A Generative Model of Terrain for Autonomous Navigation in Vegetation." *The International Journal of Robotics Research* 25(12): 1287–1304.

Wellington, Carl, and Anthony Stentz. 2004. "Online Adaptive Rough-Terrain Navigation Vegetation." In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, , 96–101.

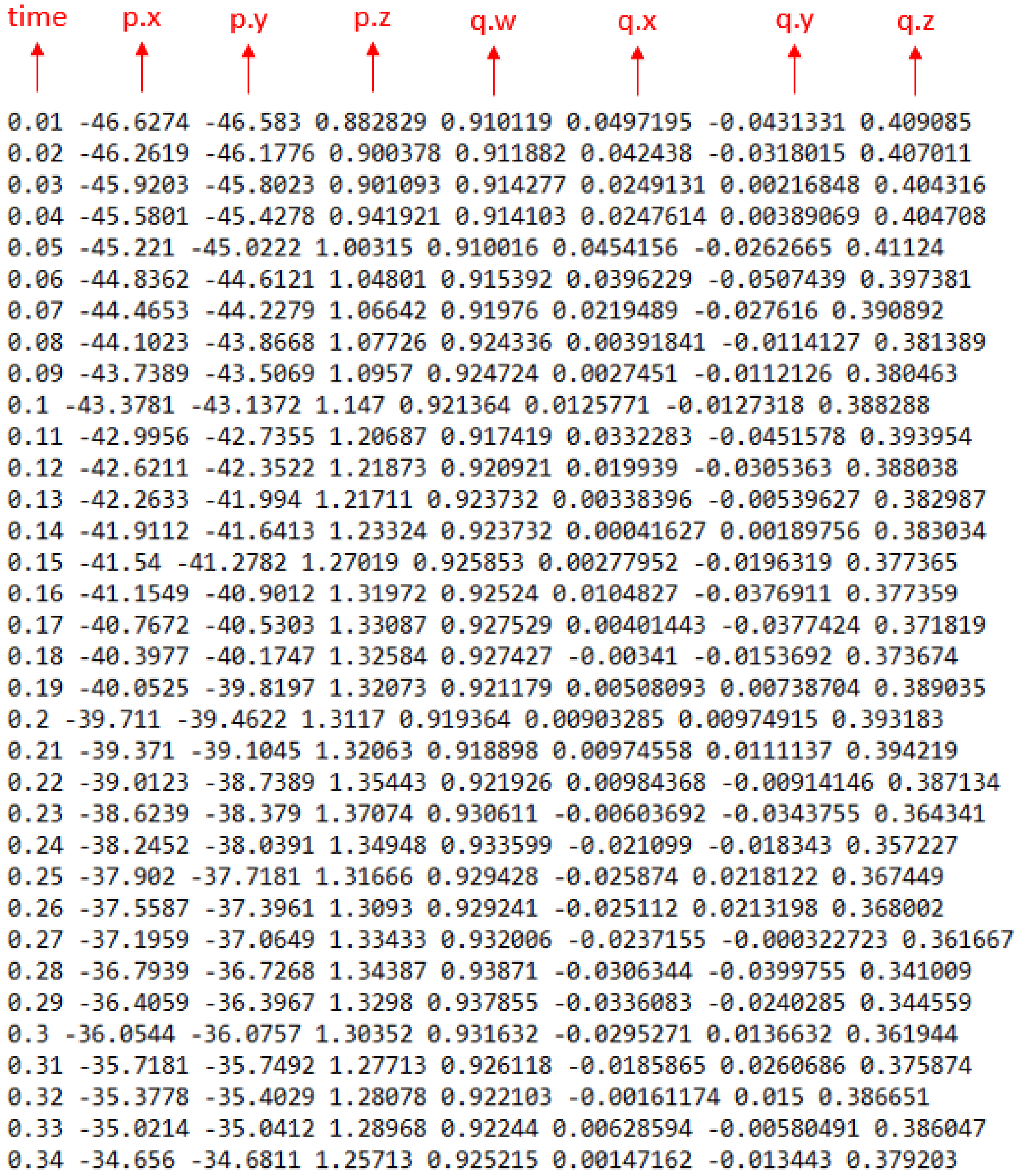
Wiemann, Thomas et al. 2010. "Automatic Construction of Polygonal Maps from Point Cloud Data." In *2010 IEEE Safety Security and Rescue Robotics*, , 1–6.

Wurm, Kai M et al. 2014. "Identifying Vegetation from Laser Data in Structured Outdoor Environments." *Robotics and Autonomous Systems* 62(5): 675–84.

Young, Parker, Sam Kysar, and Jeremy P Bos. 2020. "Unreal as a Simulation Environment for Offroad Autonomy." In *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2020*, , 114150F.

APPENDIX A

EXAMPLE DATA FILE FOR VEHICLE POSITIONS



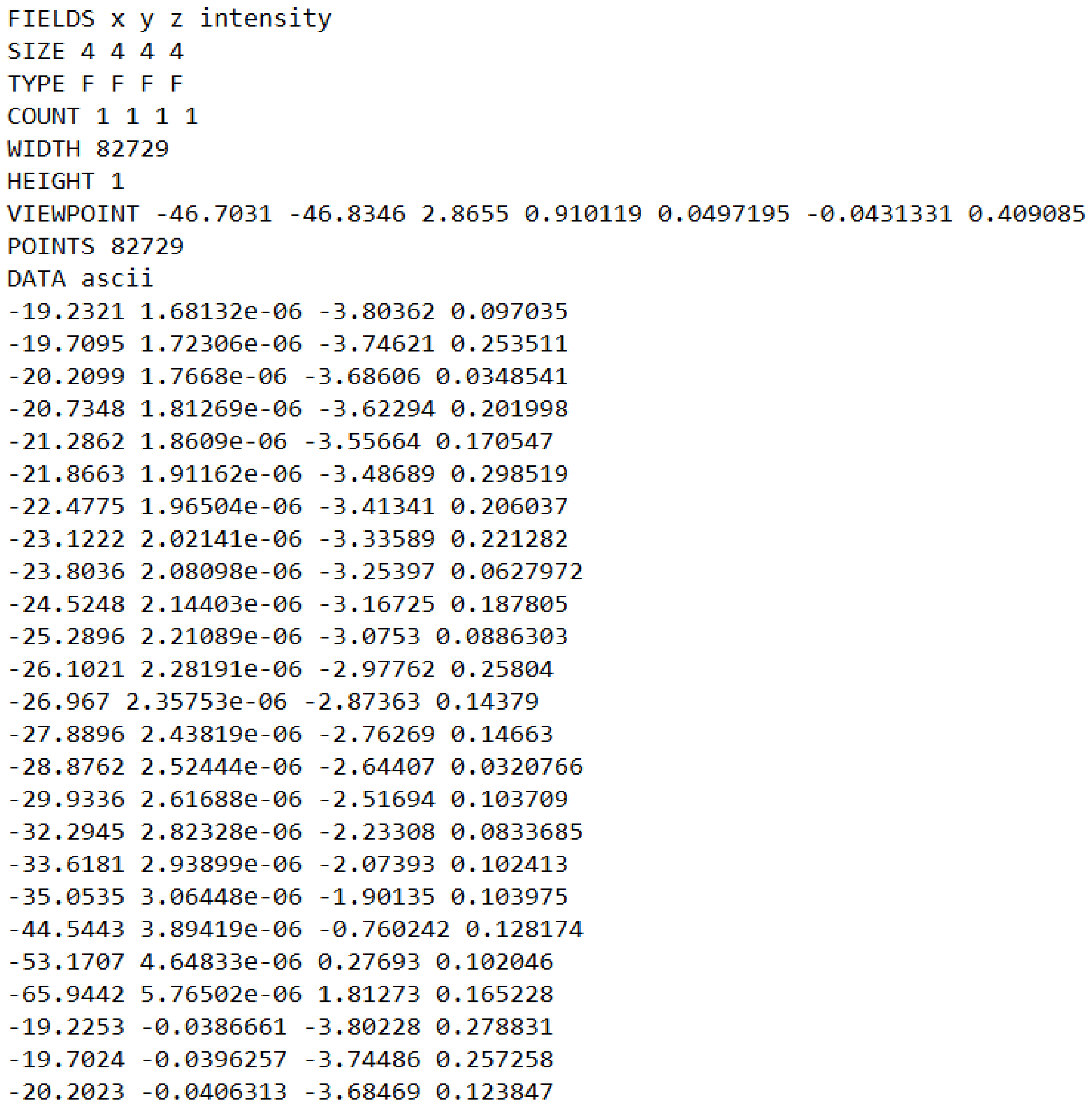
```
0.01 -46.6274 -46.583 0.882829 0.910119 0.0497195 -0.0431331 0.409085
0.02 -46.2619 -46.1776 0.900378 0.911882 0.042438 -0.0318015 0.407011
0.03 -45.9203 -45.8023 0.901093 0.914277 0.0249131 0.00216848 0.404316
0.04 -45.5801 -45.4278 0.941921 0.914103 0.0247614 0.00389069 0.404708
0.05 -45.221 -45.0222 1.00315 0.910016 0.0454156 -0.0262665 0.41124
0.06 -44.8362 -44.6121 1.04801 0.915392 0.0396229 -0.0507439 0.397381
0.07 -44.4653 -44.2279 1.06642 0.91976 0.0219489 -0.027616 0.390892
0.08 -44.1023 -43.8668 1.07726 0.924336 0.00391841 -0.0114127 0.381389
0.09 -43.7389 -43.5069 1.0957 0.924724 0.0027451 -0.0112126 0.380463
0.1 -43.3781 -43.1372 1.147 0.921364 0.0125771 -0.0127318 0.388288
0.11 -42.9956 -42.7355 1.20687 0.917419 0.0332283 -0.0451578 0.393954
0.12 -42.6211 -42.3522 1.21873 0.920921 0.019939 -0.0305363 0.388038
0.13 -42.2633 -41.994 1.21711 0.923732 0.00338396 -0.00539627 0.382987
0.14 -41.9112 -41.6413 1.23324 0.923732 0.00041627 0.00189756 0.383034
0.15 -41.54 -41.2782 1.27019 0.925853 0.00277952 -0.0196319 0.377365
0.16 -41.1549 -40.9012 1.31972 0.92524 0.0104827 -0.0376911 0.377359
0.17 -40.7672 -40.5303 1.33087 0.927529 0.00401443 -0.0377424 0.371819
0.18 -40.3977 -40.1747 1.32584 0.927427 -0.00341 -0.0153692 0.373674
0.19 -40.0525 -39.8197 1.32073 0.921179 0.00508093 0.00738704 0.389035
0.2 -39.711 -39.4622 1.3117 0.919364 0.00903285 0.00974915 0.393183
0.21 -39.371 -39.1045 1.32063 0.918898 0.00974558 0.0111137 0.394219
0.22 -39.0123 -38.7389 1.35443 0.921926 0.00984368 -0.00914146 0.387134
0.23 -38.6239 -38.379 1.37074 0.930611 -0.00603692 -0.0343755 0.364341
0.24 -38.2452 -38.0391 1.34948 0.933599 -0.021099 -0.018343 0.357227
0.25 -37.902 -37.7181 1.31666 0.929428 -0.025874 0.0218122 0.367449
0.26 -37.5587 -37.3961 1.3093 0.929241 -0.025112 0.0213198 0.368002
0.27 -37.1959 -37.0649 1.33433 0.932006 -0.0237155 -0.000322723 0.361667
0.28 -36.7939 -36.7268 1.34387 0.93871 -0.0306344 -0.0399755 0.341009
0.29 -36.4059 -36.3967 1.3298 0.937855 -0.0336083 -0.0240285 0.344559
0.3 -36.0544 -36.0757 1.30352 0.931632 -0.0295271 0.0136632 0.361944
0.31 -35.7181 -35.7492 1.27713 0.926118 -0.0185865 0.0260686 0.375874
0.32 -35.3778 -35.4029 1.28078 0.922103 -0.00161174 0.015 0.386651
0.33 -35.0214 -35.0412 1.28968 0.92244 0.00628594 -0.00580491 0.386047
0.34 -34.656 -34.6811 1.25713 0.925215 0.00147162 -0.013443 0.379203
```

Figure A.1 Sample data file for Vehicle Positions generated by MAVS

Snapshot above is an example of Vehicle Positions data. The complete file includes 272 rows i.e. 272 times the vehicle position was captured in fixed time intervals. p represents the position and q the orientation as a quaternion.

APPENDIX B

EXAMPLE DATA FILE FOR LIDAR READINGS



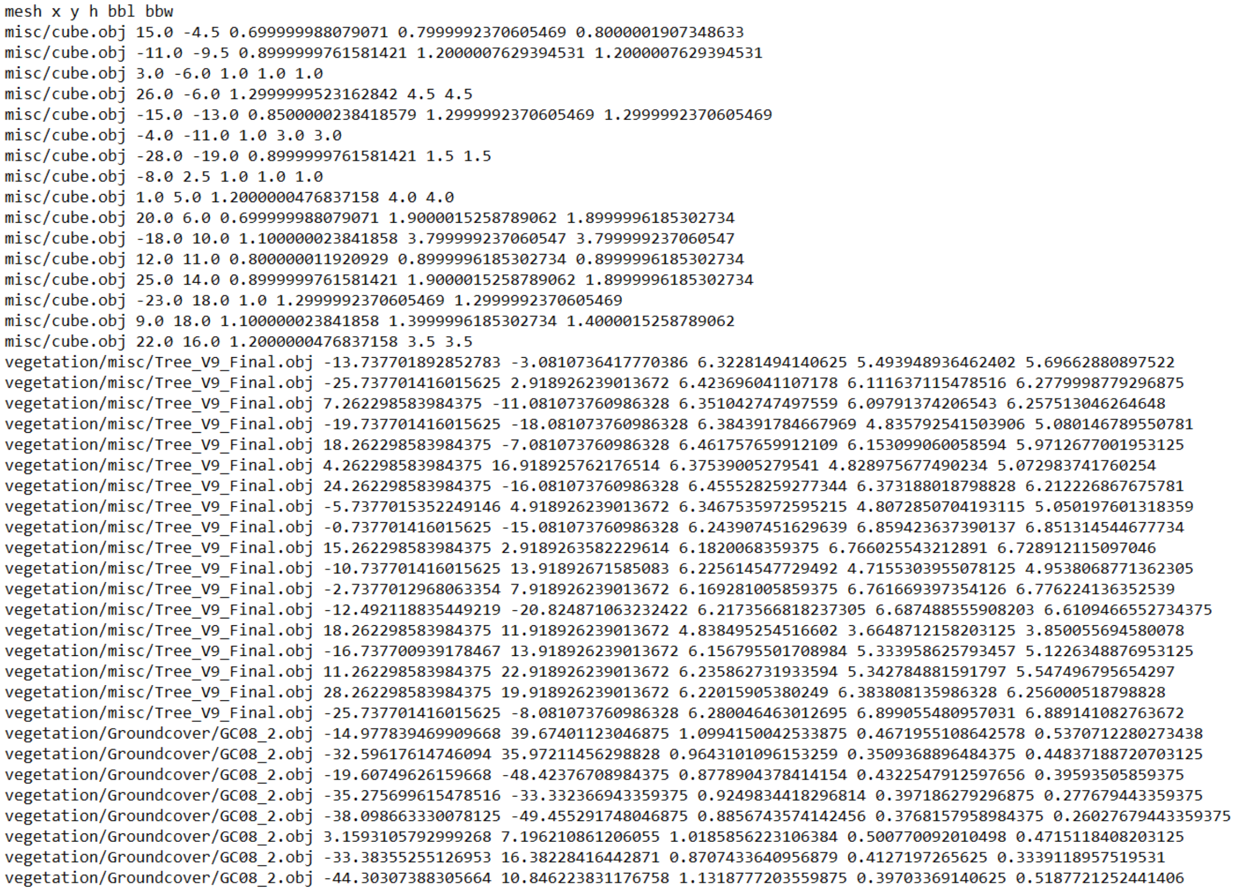
```
FIELDS x y z intensity
SIZE 4 4 4 4
TYPE F F F F
COUNT 1 1 1 1
WIDTH 82729
HEIGHT 1
VIEWPOINT -46.7031 -46.8346 2.8655 0.910119 0.0497195 -0.0431331 0.409085
POINTS 82729
DATA ascii
-19.2321 1.68132e-06 -3.80362 0.097035
-19.7095 1.72306e-06 -3.74621 0.253511
-20.2099 1.7668e-06 -3.68606 0.0348541
-20.7348 1.81269e-06 -3.62294 0.201998
-21.2862 1.8609e-06 -3.55664 0.170547
-21.8663 1.91162e-06 -3.48689 0.298519
-22.4775 1.96504e-06 -3.41341 0.206037
-23.1222 2.02141e-06 -3.33589 0.221282
-23.8036 2.08098e-06 -3.25397 0.0627972
-24.5248 2.14403e-06 -3.16725 0.187805
-25.2896 2.21089e-06 -3.0753 0.0886303
-26.1021 2.28191e-06 -2.97762 0.25804
-26.967 2.35753e-06 -2.87363 0.14379
-27.8896 2.43819e-06 -2.76269 0.14663
-28.8762 2.52444e-06 -2.64407 0.0320766
-29.9336 2.61688e-06 -2.51694 0.103709
-32.2945 2.82328e-06 -2.23308 0.0833685
-33.6181 2.93899e-06 -2.07393 0.102413
-35.0535 3.06448e-06 -1.90135 0.103975
-44.5443 3.89419e-06 -0.760242 0.128174
-53.1707 4.64833e-06 0.27693 0.102046
-65.9442 5.76502e-06 1.81273 0.165228
-19.2253 -0.0386661 -3.80228 0.278831
-19.7024 -0.0396257 -3.74486 0.257258
-20.2023 -0.0406313 -3.68469 0.123847
```

Figure B.1 Sample data file representing LIDAR reading generated by MAVS

Snapshot above is an example of LIDAR readings at one viewpoint. The complete file includes 82729 readings (82729 reflected rays). As part of data preprocessing, the first 9 rows are removed. The remaining rows contain information about each point cloud (first three columns are x, y, z distance from the LIDAR and the fourth column is the ray intensity.

APPENDIX C

EXAMPLE FILE FOR ECOSYSTEM OBJECTS



```
mesh x y h bbl bbw
misc/cube.obj 15.0 -4.5 0.699999988079071 0.7999992370605469 0.800001907348633
misc/cube.obj -11.0 -9.5 0.8999999761581421 1.2000007629394531 1.2000007629394531
misc/cube.obj 3.0 -6.0 1.0 1.0 1.0
misc/cube.obj 26.0 -6.0 1.2999999523162842 4.5 4.5
misc/cube.obj -15.0 -13.0 0.8500000238418579 1.2999992370605469 1.2999992370605469
misc/cube.obj -4.0 -11.0 1.0 3.0 3.0
misc/cube.obj -28.0 -19.0 0.8999999761581421 1.5 1.5
misc/cube.obj -8.0 2.5 1.0 1.0 1.0
misc/cube.obj 1.0 5.0 1.2000000476837158 4.0 4.0
misc/cube.obj 20.0 6.0 0.699999988079071 1.9000015258789062 1.8999996185302734
misc/cube.obj -18.0 10.0 1.100000023841858 3.799999237060547 3.799999237060547
misc/cube.obj 12.0 11.0 0.800000011920929 0.8999996185302734 0.8999996185302734
misc/cube.obj 25.0 14.0 0.8999999761581421 1.9000015258789062 1.8999996185302734
misc/cube.obj -23.0 18.0 1.0 1.2999992370605469 1.2999992370605469
misc/cube.obj 9.0 18.0 1.100000023841858 1.3999996185302734 1.4000015258789062
misc/cube.obj 22.0 16.0 1.2000000476837158 3.5 3.5
vegetation/misc/Tree_V9_Final.obj -13.737701892852783 -3.0810736417770386 6.32281494140625 5.493948936462402 5.69662880897522
vegetation/misc/Tree_V9_Final.obj -25.737701416015625 2.918926239013672 6.423696041107178 6.111637115478516 6.2779998779296875
vegetation/misc/Tree_V9_Final.obj 7.262298583984375 -11.081073760986328 6.351042747497559 6.09791374206543 6.257513046264648
vegetation/misc/Tree_V9_Final.obj -19.737701416015625 -18.081073760986328 6.384391784667969 4.835792541503906 5.080146789550781
vegetation/misc/Tree_V9_Final.obj 18.262298583984375 -7.081073760986328 6.461757659912109 6.153099060058594 5.9712677001953125
vegetation/misc/Tree_V9_Final.obj 4.262298583984375 16.918925762176514 6.37539005279541 4.828975677490234 5.072983741760254
vegetation/misc/Tree_V9_Final.obj 24.262298583984375 -16.081073760986328 6.455528259277344 6.373188018798828 6.212226867675781
vegetation/misc/Tree_V9_Final.obj -5.7377015352249146 4.918926239013672 6.3467535972595215 4.8072850704193115 5.050197601318359
vegetation/misc/Tree_V9_Final.obj -0.737701416015625 -15.081073760986328 6.243907451629639 6.859423637390137 6.851314544677734
vegetation/misc/Tree_V9_Final.obj 15.262298583984375 2.9189263582229614 6.1820068359375 6.766025543212891 6.728912115097046
vegetation/misc/Tree_V9_Final.obj -10.737701416015625 13.91892671585083 6.225614547729492 4.7155303955078125 4.9538068771362305
vegetation/misc/Tree_V9_Final.obj -2.7377012968063354 7.918926239013672 6.169281005859375 6.761669397354126 6.776224136352539
vegetation/misc/Tree_V9_Final.obj -12.492118835449219 -20.824871063232422 6.2173566818237305 6.687488555908203 6.6109466552734375
vegetation/misc/Tree_V9_Final.obj 18.262298583984375 11.918926239013672 4.838495254516602 3.6648712158203125 3.850055694580078
vegetation/misc/Tree_V9_Final.obj -16.737700939178467 13.918926239013672 6.156795501708984 5.333958625793457 5.1226348876953125
vegetation/misc/Tree_V9_Final.obj 11.262298583984375 22.918926239013672 6.235862731933594 5.342784881591797 5.547496795654297
vegetation/misc/Tree_V9_Final.obj 28.262298583984375 19.918926239013672 6.22015905380249 6.383808135986328 6.256000518798828
vegetation/misc/Tree_V9_Final.obj -25.737701416015625 -8.081073760986328 6.280046463012695 6.899055480957031 6.889141082763672
vegetation/Groundcover/GC08_2.obj -14.977839469909668 39.67401123046875 1.0994150042533875 0.4671955108642578 0.5370712280273438
vegetation/Groundcover/GC08_2.obj -32.59617614746094 35.97211456298828 0.9643101096153259 0.3509368896484375 0.44837188720703125
vegetation/Groundcover/GC08_2.obj -19.60749626159668 -48.42376708984375 0.8778904378414154 0.4322547912597656 0.39593505859375
vegetation/Groundcover/GC08_2.obj -35.275699615478516 -33.332366943359375 0.9249834418296814 0.397186279296875 0.277679443359375
vegetation/Groundcover/GC08_2.obj -38.098663330078125 -49.455291748046875 0.8856743574142456 0.376815795898437５ 0.26027679443359375
vegetation/Groundcover/GC08_2.obj 3.1593105792999268 7.196210861206055 1.0185856223106384 0.500770092010498 0.4715118408203125
vegetation/Groundcover/GC08_2.obj -33.38355255126953 16.38228416442871 0.8707433640956879 0.4127197265625 0.3339118957519531
vegetation/Groundcover/GC08_2.obj -44.30307388305664 10.846223831176758 1.1318777203559875 0.39703369140625 0.5187721252441406
```

Figure C.1 Sample data file for Ecosystem Objects generated by MAVS

Snapshot above represents an example of a data file containing types, coordinates and dimensions of vegetations and other objects in the simulated Scene 2 for chapter 6. The first column represents the type and name of the object, and the next three columns contain *x*, *y* coordinates and height of the object. The last two columns contain length and width of the object's bounding box.