MAST90107 - Data Science Project Report

Modelling Animal Movement in Forests

Group: 06

Kazi Abir ADNAN
SID: 940406
kadnan@student.unimelb.edu.au

Kaushik RAMESH
SID: 976218
rameshk@student.unimelb.edu.au

Yen-Peng CHEN
SID: 878599
yenpengc@student.unimelb.edu.au

Yuming ZHANG
SID: 973693
yumizhang@student.unimelb.edu.au

Client: A/Prof. Trent Penman
School of Ecosystem and Forest Sciences
The University of Melbourne

November, 2019

Declaration

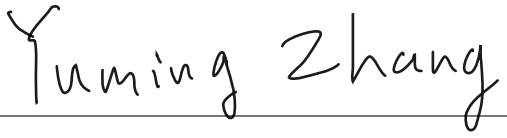
This is to certify the following:

1. This report does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person where due reference is not made in the text.
2. Clearance for this research from the University's Ethics Committee was not required.
3. The report is around 8760 words in length (excluding text in images, tables, bibliographies and appendices).

Kazi Abir ADNAN

Kaushik RAMESH

Yen-Peng CHEN



Yuming ZHANG

Acknowledgements

We would like to express our gratitude to our supervisor Professor Trevor Cohn and also Professor Howard Bondell for their support and assistance during the tenure of the project. We would also like thank our client Professor Trent Penmen for his valuable time, kind assistance and guidance throughout the period of this work. In addition, we would also like to thank kind hearted Associate Professor Michael Kirley for his invaluable thought provoking ideas which streamlined our approach towards the problem at hand. And last but not the least, we would also like to thank Faculty of Science for giving us such an wonderful opportunity to gain extended industry experience through a year long project.

Abstract

LiDAR technology plays a vital role in capturing the economic importance of forest and in policy making through monitoring forests and agricultural resources remotely. It is a prominent remote sensing technology to capture 3-D forest structure with spatial and temporal meta-data such as coordinates, colors, GPS locations, and timestamps. This project aims to use LiDAR data to recreate forest structure in 3-Dimensions to perform forest inventory analysis and object characterizations. This study also presents an automated pipeline of methodologies to identify individual trees from forest vegetation and estimate tree characteristics such as canopy heights and trunk diameter. The derived forest structure is used as an arena to simulate animal movements using Deep Q Reinforcement Learning in 2-Dimensions. Bats are chosen as the model group as they are sensitive to the density of vegetation. This paper proposes a workflow for any species of bat to take appropriate action depending on the environment by evaluating the rewards for each available action. This paper also describes a simulation-based investigation of the behavior of the movements in a controlled environment to evaluate decision-making strategies, the performance of collision avoidance, and the success rate of traveling across the forest.

Contents

List of Figures

List of Tables

1 Introduction

Forests cover about 31 percent of the world's land surface, just over 4 billion hectares [1]. They are critical resources and habitat for a vast array of plants and animals around the world. Around 300 million people worldwide live in forests, and approximately 1.6 billion depend on them for their livelihoods. Forests are not only essential for mankind but also for all the animals.They are home to 80% of the world's terrestrial biodiversity [2].

Article [3] elaborates on the importance of forest economy and the need for inventory analysis. Forest inventory analysis is attracting traction in the recent period as they provide high demand invaluable insights regarding the forest which play a vital role in formulating forest management strategies [3]. Performing an apt and proper analysis will help us in understanding the forest structure and maintain sustainable forest management. It also helps in extracting key forest properties such as tree count, average canopy height, and mean tree trunk diameter. The information may be collected manually in the field. However, due to extensive human effort, remote sensing technology is preferred.

LiDAR is an evident remote sensing technology to capture 3D forest structure with spatial and temporal meta-data, and this technology is very popular in forest ecosystem research [4]. LiDAR technology captures millions of cloud points in 3D space with high accuracy [4]. Having accurate data from the forest will help us perform efficient and precise forest inventory analysis.

This project primarily focuses on modeling animal movements in a given forest area utilizing LiDAR data. Bats are the primary animal of interest for simulating movements. The project requires a system to perform forest inventory analysis and characterize forest structure at the first stage, followed by utilizing these derived structures as an arena to simulate bats movement. Bats are chosen as the model group because they are highly sensitive to the density of vegetation.

This project addresses essential problems as it helps in understanding the forest structure as well as the living habitats of animals. As a part of the project, we performs forest inventory analysis, which can provide extensive information about the ecological structure and the dynamics of the forest for any strategic and environmental planning. In the inventory analysis, we identify various key parameters of the forest as well as individual trees. By understanding the animals' living habitat we can gain valuable knowledge about ecological behavior of animals. Besides these, forest inventory analysis also helps to address forest degradation in natural calamities or due to deforestation by providing baseline information to measure the change over the period.

Modelling animal movement is a challenging task. Animals often move across different complicated regions in the forest where human intervention is impossible. Modeling animal movement can contribute to animal ecology study revealing ecological requirements for the habitats around the forest. Furthermore, with bats being the model animal, it could assist researchers in studying their behavior and flying patterns across different types of forests.

Therefore, this work is focused on performing forest inventory analysis followed by simulating bats movement across the forest. For recreating the forest structure, we have developed an online application hosted on a cloud server. Later, the derived virtual forest structure is used as a

stage to simulate bats movement using Reinforcement Learning.

The rest of this report is organized as follows. In section 2, we divided the problem into mutually exclusive sub-problems with a predefined scope and gave a summarized overview of our approach to the solution. Section 3 contains a comprehensive description of processing LiDAR data and performing forest inventory analysis. Subsequently, section 4 describes the second sub-problem of animal movement simulation with a literature review, methodology, results, and evaluation. Lastly, the report is followed by a discussion of the methodologies, future work, major contribution of the work, and some conclusive remarks.

2 Problem Description

This study presents an automated pipeline of methodologies to perform forest inventory analysis thereupon simulating animal movement around the habitat. The first phase of the project is *Forest Recreation and Inventory Analysis* and the second phase is *Modelling Animal Movement*. Bats are the chosen model group and this adds to the challenge as there isn't any bat movement simulation data available.

2.1 Forest Recreation and Inventory Analysis

This research firstly aims to recreate forest structure from LiDAR datasets. The problem is non-trivial due to the nature of data, which has millions of unclassified cloud points with diverse forest characteristics. This demands to process and understand the data in a well-defined unsupervised efficient fashion. In this part, the task is to study LiDAR data and identify individual trees with their position from forest vegetation and estimate tree characteristics such as canopy heights, trunk diameter, along with characterizing the shape of trees. The main objective of this work is to develop a 3D representation of the forest and to characterize each object. The process of recreating the forest structure involves several algorithms, such as terrain and vegetation identification. Then, we are segmenting vegetation into individual trees and obtain not only its position but also its characteristics such as height, canopy height, diameter at breast height(DBH), and crown height. All these estimated parameters represent the forest structure, aiding to perform forest inventory analysis.

2.2 Modelling Animal Movement in Forests

In the second phase, the project focuses on simulating bats movement around the characterized habitat. The goal is to simulate the bat movement patterns across a characterized forest F. In order to approach the problem, we make a few assumptions such as bat flies at a particular constant height in the forest with a constant velocity and the size of the bat is fixed as well. In this problem, We are not considering any environmental and temporal factors as well as any aerodynamics to simulate the movement. Due to the lack of real bat movement data, we framed the simulation problem as a Reinforcement Learning task.

The interaction between bat and the forest can be naturally modeled as an agent and its surrounding environment. Consequently, the problem of simulating the bat's movement through the forest can be defined as training an agent that is able to fly through an environment while avoiding obstacles on the way. More concretely, given a 2-Dimensional representation of the forest structure encoded as a matrix of 0/1 values where 1 represents obstacle and a randomly selected coordinate point $A(x_1, y_1)$ at one end of the forest structure as a starting point of the bat, we would like to train a model to learn proper turning angles to avoid the obstacles and reach the other end of forest. Figure 1 is a graphical representation of the defined problem.

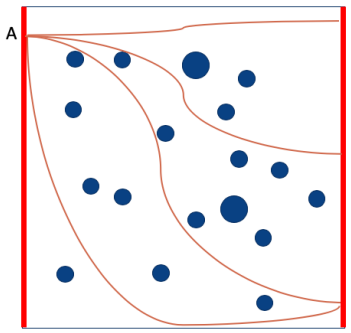


Figure 1: Bat Movement Abstract

3 Forest Recreation and Inventory Analysis

This section covers the literature review, step by step methodologies, and corresponding results of forest inventory analysis in the following subsections.

3.1 Literature Review

In the last two decades, the LiDAR system gained its popularity in different domains [5] and is widely used because of its accuracy and precision. The data collected by the LiDAR system over large forest areas is an efficient and practical alternative to time exhaustive and expensive field surveys. The data is composed of XYZ coordinate points with spatial and temporal information of a target area [6]. Therefore it is necessary to understand the state of the art algorithms to comprehend the characteristics of LiDAR data, various processing techniques, and methods to extract useful information. The works of [4, 5] demonstrates a complete work-flow to recreate 3D forest sites using terrestrial LiDAR data.

In literature [4–6], LiDAR has been used as a primary source of data, and they have almost carried out similar tasks to the proposed first phase. [4,5] proposed a framework to automatically detect trees, which includes several pre-processing steps, helping in understanding the required processes prior to classifying the trees. These led to transform the cloud points of the forest data to height-normalized data for canopy height measurement. These papers also include methodologies to classify the crown of a tree and the Diameter at Breast Height (DBH), which has been inherited in the first phase of the project. This project also requires the classification of data points to identify trees and estimate its properties such as DBH, height and canopy area. In the article [7], several algorithms were found which are related to tree segmentation and the estimation of DBH that has been used to extract the tree properties.

Several computational tools have been explored to process LiDAR data. [4] developed an open source software tool *3DForest* to analyze Terrestrial LiDAR data for tree segmentation, visualization, and calculating trees parameters. However, the program was unable to load

large LiDAR data files(sizes above 300MB). [8] proposed LASTools library for LiDAR data processing, which is designed primarily for airborne data. It is available for licensing to prevent the distortion in the processed data. Lastly, [9] an open source software that can be used to create plugins and estimate forest inventory parameters, which depend on LASTools to process LiDAR data. Finally, we ended up using R programming language as our main tool for development due to the availability of relevant libraries to analyze forest structure and process LiDAR data. Couple of useful libraries *lidR* (well-known library for the analysis of airborne LiDAR data) and *TreeLS* (library designed for terrestrial LiDAR data) are available to extract certain tree characteristics. None of the works in literature or the tools are self-sufficient to fulfil the requirements of the first phase. Eventually, the proposed work is a perfect mix of all available latest work for forest inventory analysis to the date.

3.2 Dataset

LiDAR(Light Detection and Ranging) data is provided to understand and recreate forest structure. It is a 3D remote scanning technology that has enabled accurate and precise measurement of 3D spatial forest structure with possible classifications of high, medium, or low vegetation, etc. Data is collected from airborne platforms named as Aerial Laser Scanning (ALS) and from ground based remote sensing technique named as Terrestrial Laser Scanning (TLS). The Table 1 shows the properties of the provided Aerial and Terrestrial LiDAR data. Data is provided in *'las'* and *'e57'* format which hold similar information.

Features	**Aerial**	**Terrestrial**
X, Y, Z coordinates	✓	✓
RGB Color	×	✓
Classification	✓	×
Intensity	✓	×
GPS	×	×
Timestamp	×	×

Table 1: Properties of existing LiDAR data

3.2.1 Aerial Data

Figure 2 shows a sample of Aerial LiDAR data of a forest site. The different colors in the data represent different classes of vegetation. The application of aerial LiDAR data to forest inventory analysis has shown to be effective for forest object characterization and extraction of forest inventory parameters [5].

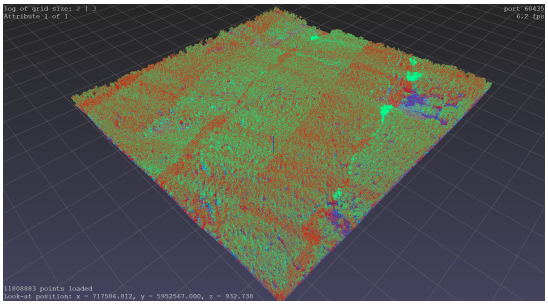


Figure 2: Aerial LiDAR overview

3.2.2 Terrestrial Data

A sample terrestrial LiDAR data is shown in the figure 3. The use of terrestrial scanning techniques often offers precise and detailed information that is not visible from an aerial perspective [5]. Indeed this is a powerful technology to represent the 3D structure of forests with a high level of detail and accuracy. Many algorithms have been proposed recently for forest inventory analysis using TLS data, which demonstrates the increasing use of this technology [4].

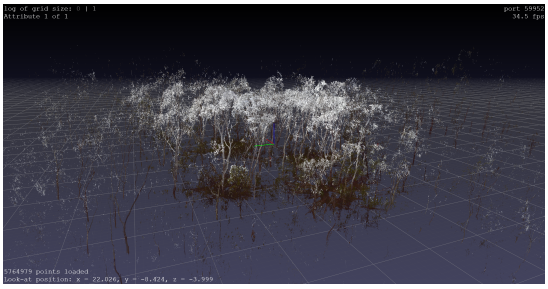
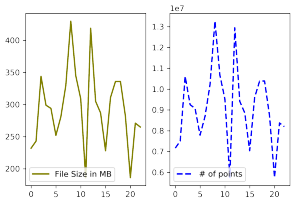
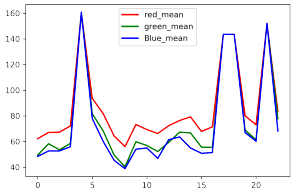


Figure 3: Terrestrial LiDAR overview

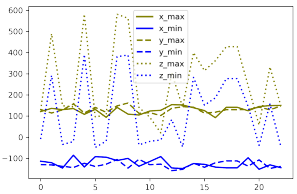
3.2.3 Analysis Of Data



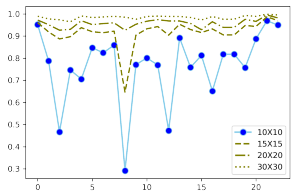
(a) Size of files



(b) Mean of RGB values



(c) Min-Max values of X, Y, Z



(d) Density of cloud points

Figure 4: Analysis result of Terrestrial LiDAR data

Terrestrial LiDAR data has been predominantly utilized for simulation of animal movements as it holds detailed information of the forest. Preliminary analyses were performed on 25 sites of terrestrial data. The figure 4a displays the range of file sizes and the number of cloud points in each of the sites. We found that the file sizes mostly range between 200MB to 500MB, and the average number of points in file is up to 10 Million. As we already know that terrestrial LiDAR data barely holds any RGB information, the figure 4b confirms it as the mean RGB value is less than 60 in most of the cases. Scanning range of ground scanner was around 100m in both x and y direction which is shown in the figure 4c. However, there are outliers along the z axis, which reaches to 600m for some sites. It is possible that the LiDAR scanner picks a bird or any other other object that is flying in the sky. Finally, from the Figure 4d, we can understand the density of cloud points across several grid sizes. The density of cloud points is higher near the scanner with a grid size of $30X30m^2$ and almost 95% of the data is retained. The further away the points are from the center, the cloud points become sparse.

3.3 Methodology

This section explains the process of recreating the forest structure using raw LiDAR data. The process involves data preprocessing and usage of different modelling techniques to represent the ecological structure.

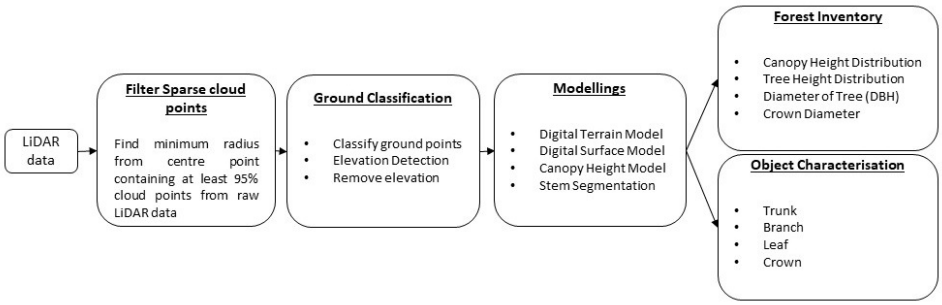
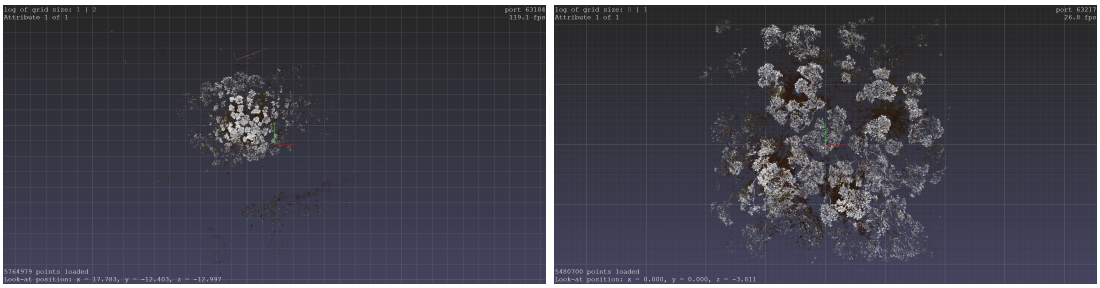


Figure 5: Data Processing Pipeline of Forest Inventory Analysis

3.3.1 Filtering Sparse Cloud Points

Due to the nature of how terrestrial LiDAR data are collected, cloud points close to the centre of scanner have higher density. Objects formed in these regions have dense points compared to moving away from the centre. In order to obtain the best representation without losing much information, we filtered out the sparse region by specifying the distance from the centre. This distance is chosen based on the desired percentage of cloud points to retain, which depends on sites and typically ranges between 15 to 40 meters from the center.



(a) Full Carpark site (b) Filtered Carpark site

Figure 6: Filtered Data by $16m$ radius in x & y axis from center (contains 95% of original data)

3.3.2 Removing Ground Points

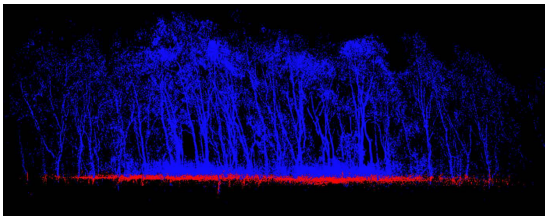
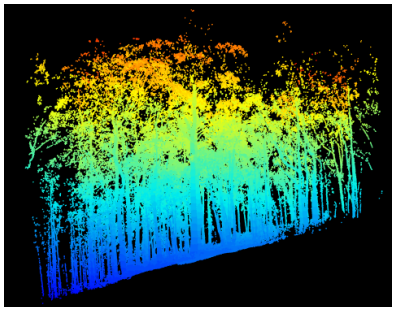


Figure 7: Filtering Ground Points

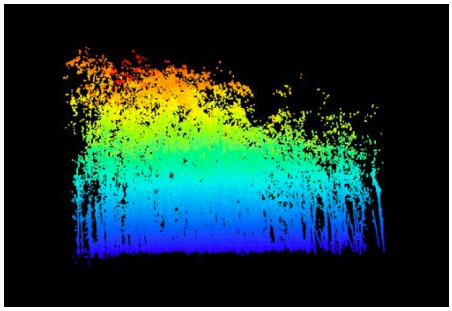
Next, we classified each of the cloud points as either *"ground"* or *"vegetation"*. The primary reason is to remove the ground points to reduce the computational cost. The points classified as ground are less likely to be relevant for tree parameter detection. Secondly, the sets of ground points are required as input to calculate the Digital Terrain Model, which is utilized in elevation detection.

3.3.3 Elevation Processing

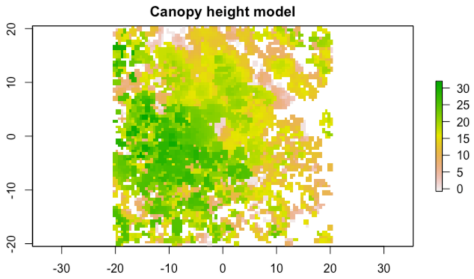
This step identifies and flattens the elevation of terrain in the given data to get the canopy height. Digital Terrain Model (DTM) is the elevation of the Earth's surface. By normalizing the level of original cloud points with computed DTM, we can flatten the cloud points. Figure 8a is a site with elevation, figure 8d is the DTM computed from it. As can be seen in figure8b , the resulting point cloud is flattened after subtracting the level of DTM from the original point cloud. The flatten cloud points are then used to compute the Digital Surface Model (DSM). DSM, in this case, is actually the Canopy High Model(CHM), which measures heights of objects above the ground. Figure 8c depicts a CHM in 2D, the metric on the right shows the elevation level with respect to the DTM computed.



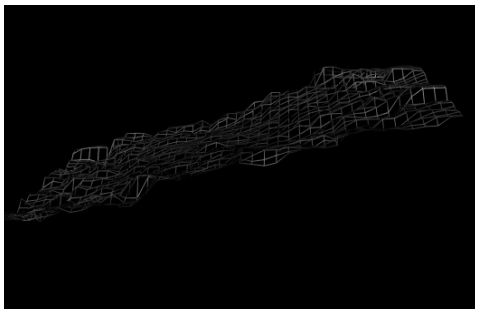
(a) Site with elevation



(b) Flatten terrain



(c) CHM in 2D



(d) Digital Terrain Model

Figure 8: Overview of Elevation Processing

3.3.4 Identifying Individual Trees

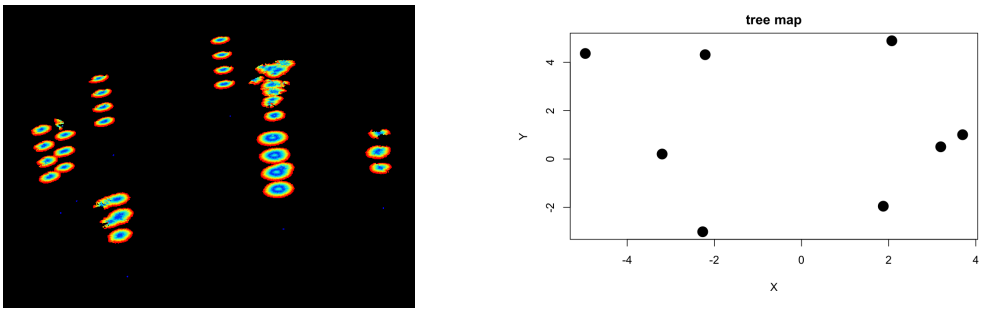
In this step, we identify individual trees and their location to estimate the number of trees in a given site. To achieve this, we follow two approaches, namely Tree Segmentation and Hough transformation. In Tree Segmentation, we have tried three different algorithms to compare and contrast the outputs. The first one is the algorithm proposed by *li2012* [10]; the image derived from this algorithm is the best among the approaches. However, the processing time grows exponentially with the number of data points, which makes it suitable only for small regions. The other two algorithms are *Dalponte2016* [11] and *Silva2016* [12]. In contrast to *li2012*, both of these are raster based algorithms. These algorithms require the precomputed Digital Surface Model(DSM) and set of identified treetops, which raises the issue of computing DSM and treetops. One of the challenges we encountered using raster based algorithms is the

parameter tuning to compute the DSM. The smoothness of the raster layer parameter and the size of the moving window parameter (used to search local maxima in the treetops detection) played an important factor in determining the number of treetops (Figure 10).

The second approach is *Hough* transformation [7], which searches for primitive shapes (like, lines or circle) on a raster data. The algorithm for tree trunk detection implements Hough Transformation adapted to find circular shapes on a two-dimensional horizontal plane [7].

3.3.5 Tree Diameter at Breast Height

In order to estimate the diameter of trees at breast height (DBH) *Hough* Transform algorithm [7] is utilized. The algorithm identifies the tree trunk by searching for a circle like pattern in the LiDAR data and returns the radii of the identified pattern around the breast height region.



(a) Estimated Tree Trunk Radii

(b) Estimated tree locations

Figure 9: Tree detection via Hough algorithm

3.3.6 Canopy Height Measurement

Using the results from the previous methods, we could derive a set of treetop values and then calculate an average value of these treetops as an estimate for the forest canopy height for a given site. As we have detected and removed the elevation of the site, the estimates of canopy height, as well as individual tree height would be equitable to the actual measures.

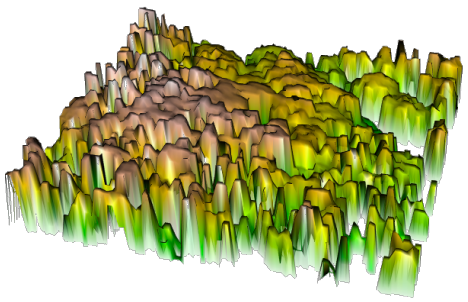
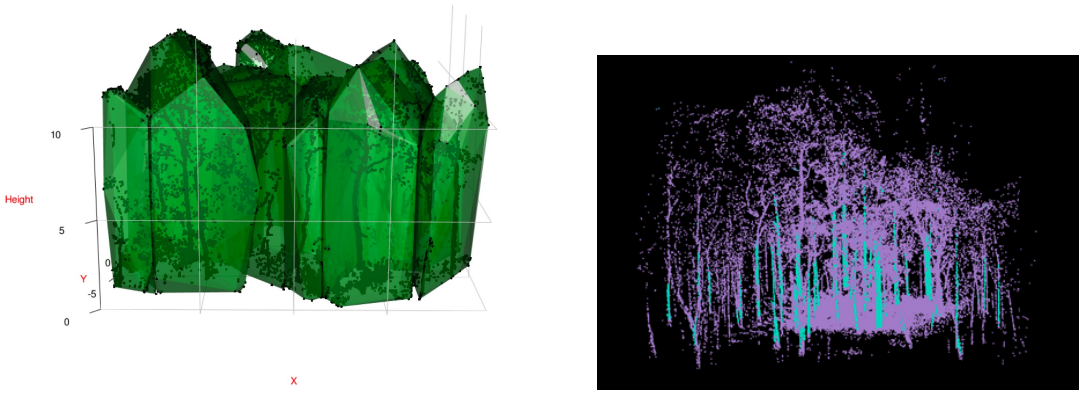


Figure 10: Canopy Height Model in 3D

3.3.7 Object Classification

Object classification is the final process of phase 1. This step bridges the two phases of the project. Under this step, we identify individual trees as well as its components to determine the obstacle for animal movement in the next phase. Multiple approaches are implemented to classify the objects, namely stem segmentation and convex hull detection of the tree. Stem segmentation recognizes the stem/trunk of trees in a given site wherein the latter constructs a convex hull for the entire tree structure. Stem segmentation approach is preferred over the other

as the resultant obstacles from the convex hull detection are quite dense, leaving no available paths to traverse across the forest. On the contrary, the obstacles from stem segmentation are sparse. Hence a convex hull of a $5m$ to $8m$ cross section of stem segments are obtained from the data as a reasonable approximation of the obstacles in the forest.



(a) Convex hull of trees

(b) Tree Stem Segmentation

Figure 11: Object Characterization of tree

3.4 Online Application

We developed R-Shiny application in order to perform the forest inventory analysis. The application forms the pipeline to perform the analysis with multiple sections, each having its own features and results. The above discussed methodologies are implemented and evaluated on multiple LiDAR forest data. The advantage of this application is that it is modular and can perform the required analysis on any new LiDAR data. It is hosted on Melbourne Research Cloud [13] server and currently being used by the School of Ecosystem and Forest Sciences at the University of Melbourne. Figure 12 displays the home page of the application, where the results of the analysis are displayed for the chosen site.

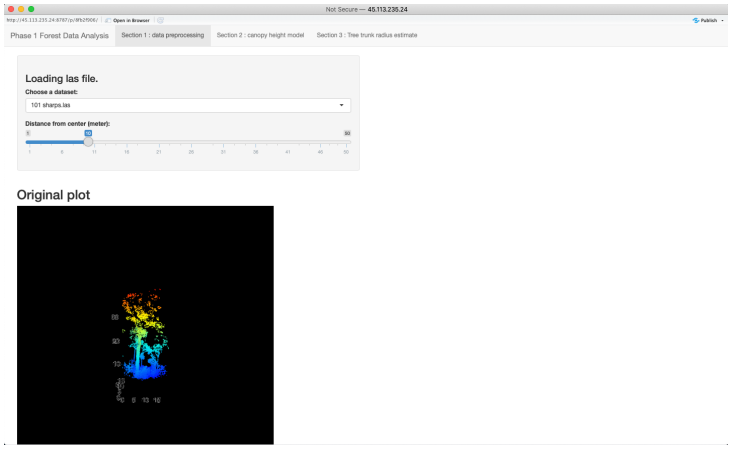


Figure 12: Online Shiny Application

Site name	Using Tree-top	Using cross-section	Manual count
84ish seaview.las	13	16	16
85 seaview.las	5	5	5
94 seaview.las	5	5	5
95 seaview.las	3	3	3
SP02 big hill.las	4	3	4

Table 2: Number of trees comparison

3.5 Result Analysis

In this section, we analyse the results obtained from the application.

3.5.1 Tree Count Measurement

One of the key forest parameters is the number of trees in a forest. In order to derive a reasonable estimate, we implemented multiple algorithms aforementioned. The results from both the algorithms for a handpicked number of sites are tabulated below, along with the manual counts (table 2). Figure 13 shows the tree count estimates from both algorithms for all the sites. The estimates are fairly consistent across the algorithms with minor discrepancies due to the tuning parameters involved.

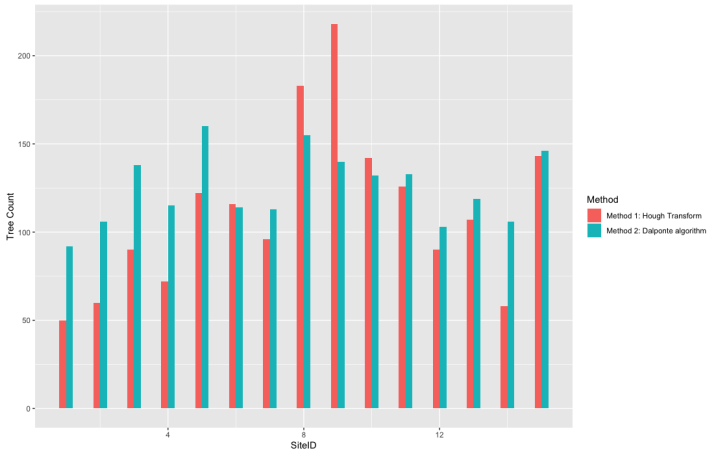


Figure 13: Tree counts at different sites

3.5.2 Tree Height Distributions

Using the application, we obtained the individual tree height for each of the sites and created a tree height distribution plot (figure 14). From the plot, we can understand that the height of the trees ranges from $10m$ to $30m$ on average across different sites. This is in accordance with the Australian Forest Profiles provided by the Department of Agriculture and Water Resources [14]. This confirms that the measurement is a reasonable estimate of the actual tree height.

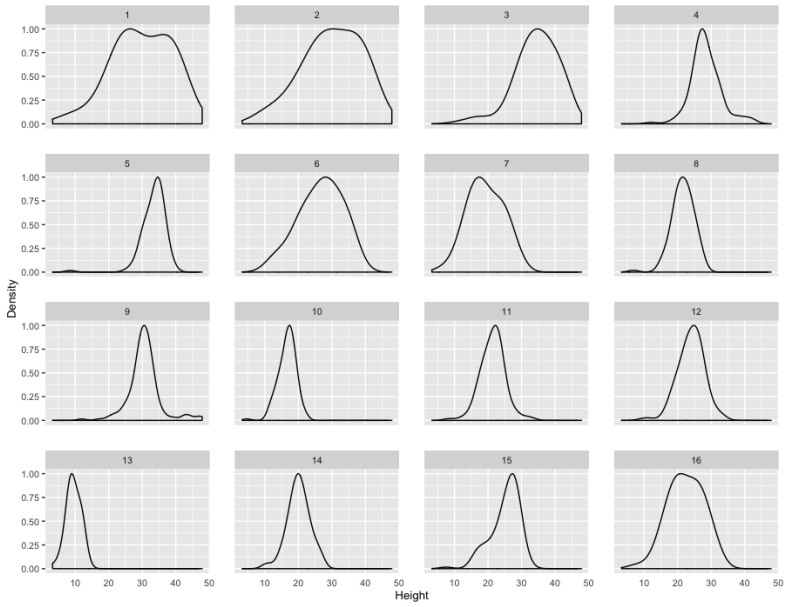


Figure 14: Tree height distribution at different sites

3.5.3 Diameter at Breast Height Distributions

Diameter at breast height distribution (DBH) is one of the key forest economics. The application is capable of estimating the DBH of each of the trees for a given site. Using the estimates, we plot a diameter distribution (figure 15). The plot clearly shows that the diameter ranges from $0.40m$ to $0.45m$. According to the Andrew Haywood and his team [15], the mean DBH across different bio-regions ranges between $0.25m$ and $0.45m$. On comparing the results, we understand that our DBH estimates are admissible.

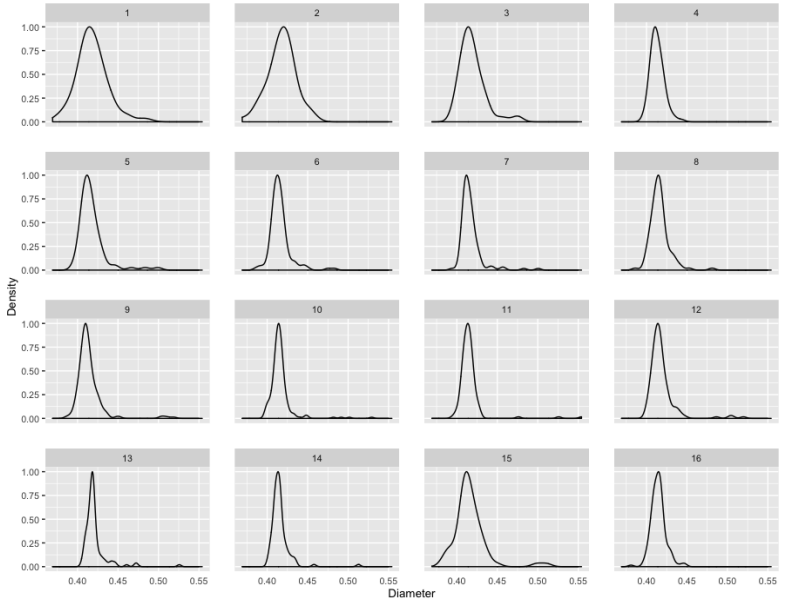


Figure 15: Tree diameter distribution at different sites

3.6 Evaluation

One of the key challenges of the project is to evaluate the results of each of the outcomes as there were no supervised data available to verify the outputs. Hence, we resorted to find relevant works to compare the results. In order to estimate the tree counts, we implemented multiple algorithms to cross evaluate, and manually counted the trees for a set of sites with small window size and the results are as tabulated in table 2. Using the results, we evaluated the output of the algorithms and observed that the results are stable and uniform with minor discrepancies. For the tree height and DBH metrics, we tried identifying information provided by trusted organizations as well as relevant research works. The forest profile information provided by the Australian government helped in evaluating these metrics, and we could understand that the results are reasonable estimates of the metrics. However, it is important to note that tree height and diameter depends on the type of tree in the forest. In our study, no detailed information about tree types in forest data is available. On the other hand, [16] evaluated the result of estimating the tree characteristics categorizing trees based on age, tree species, and other critera using a labelled dataset.

4 Modelling Bats Movement

Sensory information plays the most important role in any animal movement around a forest as it helps in finding objects of interest, such as food, shelters, predators, and migration. If we talk about Bat, sensory abilities are limited for many species and mostly rely on echolocation information, which defines their ecological behavior. In addition, bats are nocturnal and mostly flies to reach destination facing numerous perceptual challenges with limited sensory information. Therefore, to simulate the complex behavior of bats movement, it is necessary to understand the different kind of species and their technique of movement. It is worth to note that the kinematic movement of bats is out of the scope of this study.

4.1 Literature Review

Articles [17–19] characterized the habits and behavior of bats helping to understand their habitats. In addition, they provide some bats flying parameters which can help to simulate bat movements such as flight altitude, flight angle, and flight distance. With this information, it is easier to understand bats' ecological behavior and movement objectives. Additionally, those papers also provided an understanding of different species of bats and their shapes, their differences in flying parameters, which might be helpful to design species-specific simulation. Furthermore, the mentioned papers give us an idea of navigation patterns, and the purpose of bats fly.

We have reviewed different types of algorithms to simulate bats' movement in our project which includes meta-heuristic algorithms like Bat Algorithm [20], Particle Swarm Optimization [21] and afterwards, Deep Reinforcement Learning [22]. [23] proposed *Robat*, the first fully autonomous artificial bat biologically inspired, which moves based on echolocation sensory in-

formation. It identifies the size, shape, and type of the objects and estimates the free path. However, the work is focused on designing the artificial bat-like robot. This work motivated to identify object type while moving through the forest.

Article [20] introduced Bat Algorithm (BA) to simulate the movement of Bats. [24] introduced the modified Bat algorithm based on Swarm Intelligence, which is inspired by bat behavior. Besides this, they modified the original BA with threats added in the environment to find the optimal path avoiding threats. However, the algorithm is more about finding the optimal path using swarm of bats. In our case, the interest is to learn movement individually. More importantly, the bat algorithm is focused on velocity and the position that a bat is at. In addition, it uses the idea of echolocation of the bat, and we do not have any data for the real bats. Therefore, the Bat Algorithm is not the best way for us to approach the problem.

Article [25] also provides some information about swarm optimization using Bat Algorithm. The Goal for this algorithm is to find the shortest path between a starting point and the ending point within an environment with obstacles with a group of bats moving together in the forest. On the other hand, our project is focused on how an individual bat can go through the forest. [26] proposed Glow-worm Swarm Optimization(GSO) for Path-Planning with multiple agents, which is also inappropriate for our case study due to lack of bat specific data.

Finally, we shifted the focus to design the problem using Reinforcement Learning. Using an Inverse Reinforcement Learning framework, [27] have developed a novel method to predict the most likely route that an animal would have traveled. Within this, an algorithm learns a reward function from animal trajectories using the features from a specific environment. This work does not help directly in our case as the focus of that work was to fit trajectories of the data that was available to them. [28] introduced Reinforcement Learning using Deep Learning network. Reinforcement learning is an algorithm that checks how agents take actions in an environment and keeps track of the performance [22]. Using Deep Q-learning may help us to train the bats and then test the algorithm in new environments. We have implemented this algorithm in our project. The reason being it fits the requirement of our project, and does not need a group of bats and works well for training a single bat.

4.2 Methodology

This section explains the process of the simulation of bats movement around the forest. The process involves data preprocessing step, learning, and evaluating the model, which will be explained subsequently in detail.

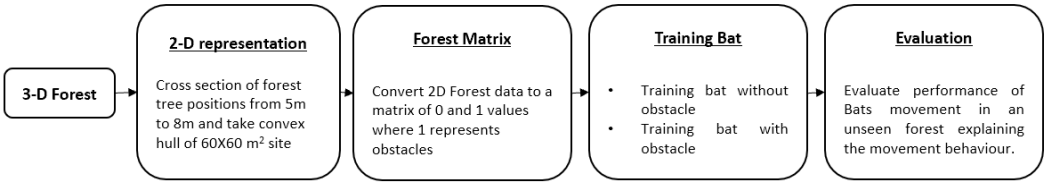
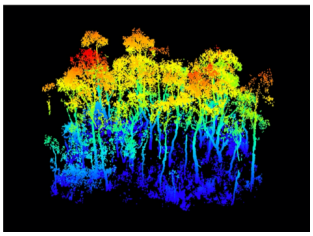


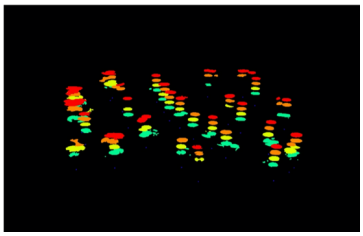
Figure 16: Key steps of Bat movement simulation

4.2.1 2D Representation of Forest

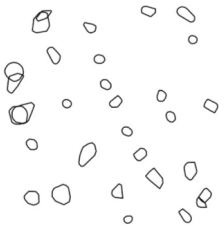
This first step of the training process is to prepare the environment where a bat would learn to fly. In this work, the simulation of bats' movement has been defined in 2-Dimensions. The figure 17b is a cross-section of trees for the site 17a from $5m$ to $8m$. Following which, we take the convex hull of the cross-section of trees as obstacles in the forest as shown in 17c



(a) $20X20m^2$ window of carpark site



(b) Cross-section of tree positions from 5m-8m

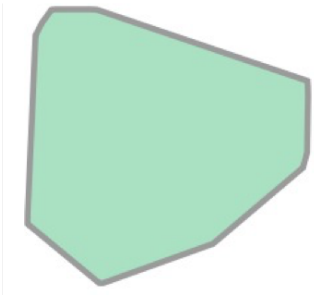


(c) Convex hull in cross-section

Figure 17: 2-D Forest representation of $20X20m^2$ window of carpark site

4.2.2 2-D Forest Matrix

We created a 2-D Forest Matrix of values $0's$ and $1's$ from the output of the first step. If we have a Polygon like 18a, then we need to transform the polygon into a matrix with 1's and 0's, where 1's represent the polygon and the 0's represent the empty spaces. Bats wont be able to fly through the polygon as it represents the obstacle. The forest 2-D matrix, 18b will be used as an input for the training phase.



(a) Convex Hull

```
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0],
 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0],
 [0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
```

(b) Corresponding Convex Hull Matrix

Figure 18: 2-D Matrix Representation

4.2.3 Training a Bat

This section covers the methodology and the assumptions made to train a bat to fly across the forest avoiding the obstacles.

4.2.3.1 Reinforcement Learning

To simulate the bat movement through the forest without any actual bat data, we approached the problem using Reinforcement Learning (RL). In RL, a problem has four major components such as an agent, environment, action, and reward. The agent takes an action based on the learning from environmental interaction to achieve a specific goal. Agent tries to learn something and the environment is the outside component where it interacts [29].

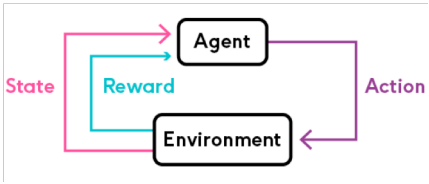


Figure 19: Reinforcement Learning Framework

The quality of action, A_t at time t depends on the agent's state S_t. That is, the quality of an action is a function of both the state and the action itself. In particular, the agent should choose the action that maximizes the expected discounted future reward. [29]. Here we used the notation G_t to denote the discount future reward stating at time t with the assumption that our learning task has a finite time horizon, T.

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+2} + \cdots = \sum_{k=0}^{T} \gamma^k R_{t+k+1}$$

Where, $0 \leq \gamma \leq 1$ denotes the discount rate for the future reward. The quality, q of an action a in the state s under some policy π (a mapping from state to action) is denoted as:

$$q_\pi(s, a) = E_\pi[G_t | S_t = s, A_t = a]$$

Estimating $q_\pi(s, a), \forall a \in A, \forall s \in S$, where A denotes the action space and S for state space, is the central idea behind one of the most basic algorithms in RL the *Q-Learning* [29]. In *Q-Learning*, the goal is to come up with an efficient algorithm to estimate $q_\pi(s, a)$. The most common approach involves using dynamic programming to estimate those *q-values* in a tabular fashion, and the resulting output is a *q-table*, in which each entry denotes a q-value for each state-action pairs.

However, the size of the q-table is the product of the size of action space and state space. In most real-world problems, the state space would be intractably large. The memory required for the large table together with the computing resource needed to estimate value for each entry would be enormous. Moreover, if state space is encoded by continuous values, there would be infinitely many possible states an agent could be in, in which even with a large number of simulations, almost all states encountered would have never been seen before [29].

4.2.3.2 Deep Neural Network (DNN)

The neural network is a class of extremely versatile non-linear function approximation. It is widely used in applications of computer vision, natural language and processing and areas requires a complex and sophisticated model. Furthermore, DNN has been responsible for many of the most important achievements in the Machine Learning system. One of the desirable properties of DNN is that the user does not need to hand-craft features as it automatically discovers the representation needed for feature detection of any learning problem [29]. Therefore, in this work we adopted DNN to estimate the optimal q-value.

$$q_\pi(s, a, w) \approx q_\pi^*(s, a)$$

Where optimal state-action value q_π^* is maximum achievable among all policies.

$$q_\pi^*(s, a) = \max_\pi q_\pi(s, a)$$

4.2.3.3 Agent and State Representation

We need to impose some assumptions to simplify the problem of simulating bats movement. These may seem imperfect but helps to formulate the movement simulation problem.

- Bat is modeled as 20cm $*$ 20cm square object.
- Bat flies at a constant velocity, moves and rotates in 2-Dimensions.
- Bat's turning angles are ranging from -20 to 20 degrees.
- Bat's movement is modeled by a discrete-time model, and the change of Bat's position as the result of each move is modeled as a vector addition in a 2-Dimensional space.
- Bat flies at a certain height, and it never changes.
- There are no wingspan movements of bats.

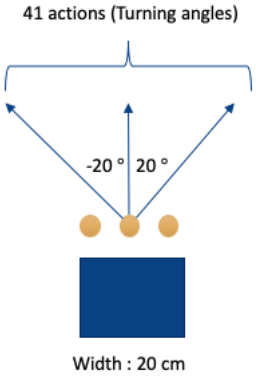


Figure 20: Agent Bat abstract

There has been some amazing achievement on DQN on tasks that are similar to the defined problem. For instance, [28] developed algorithm using DQN to train an agent to Atari game

and achieved a superhuman level, directly using only image pixels as input. This influenced us to incorporate image pixels as part of our agent state (derived from Forest matrix). However, we have decided not to do so, on the ground that it is difficult to justify image pixel as part of the bats' behavior to understand its environment.

In real world, bats sense the environment using sensory information like Echolocation. Therefore the focus shifted to compute obstacle densities around the bat using some sensors. In our design, each bat has three sensors situated at the front, each separated by 30-degrees as depicted in 20. Obstacle Density (D_i) is denoted as the ratio of points has obstacle around the sensor. These are denoted as $D_i, i = 1, 2, 3$ where:

$$D_i = \frac{\text{number of 1's in surrounding}(D_i)}{\text{number of cells in surrounding}(D_i)}$$

Where surrounding(D_i) denotes the the 25 * 25 square with center being Di.

In addition to these information representing the obstacle densities, we have also incorporated the orientation of the agent, which is the angle between the agent's facing direction and the direction towards the goal. The orientation would be positive if the agent is facing towards target. Moreover, the negative orientation is also added as part of state representation as it improves the training. These five features combined, encoded as real numbers represent the agent's state (s_t at time t).

$$s_t = \{D_1, D_2, D_3, \text{orientation}, \text{-orientation}\}$$

4.2.3.4 Training DQN

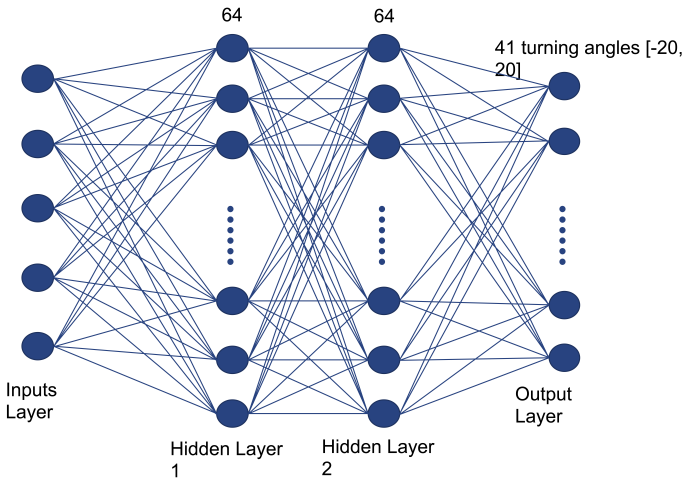


Figure 21: Neural Network Architecture

In the majority of successful deep learning applications, a huge amount of labeled training data is typically required. In contrast to that, RL algorithms agents typically learn independently from scalar rewards that are often spare and delayed [28]. For example, in the game of chess, the reward is only an award when the agent wins the game. This imposed a huge problem on the training, as it slows down the learning as the result of some lengthy and uninformed

exploration trajectories [30]. To overcome this, we have used additional reward signals in the hope that it would help to shape the agent's behavior. These rewards are given as the result of action taken where positive rewards encourages good behavior and negative rewards discourages to not to take that specific action.

- -1 if hit the tree.
- -0.3 if movement does not have any useful impact.
- 0.2 if the new position has shorter distance to goal.
- -0.9 if the new position is outside the boundary. (To discourage the bat to travel outside the forest site.)
- 10 if reach goal.

Another prominent issue while training a DNN in an RL setting is that the sequence of states $\{s_1, s_2 \ldots . s_T\}$, which are also the input samples to the DNN are highly correlated. This violates the independent samples assumption that many stochastic gradient-based algorithms rely upon, and DNN is one of those [31]. To address this issue, we have used a method called *Experience Replay*, of which we store the agent's experience at time t, denote as four tuples $e_t = (s_t, a_t, r_t, s_{t+1})$, into a fixed size queue called *replay memory*. We set the size of replay memory to be of 1000, thus at any given time, it would contains experience back to early as past 1000 steps.

Afterwards, we randomly sample a batch-size of 100 instances to perform the weight update of the DNN. There are several advantages of employing the method of experiment replay in training. Firstly, the enabling of using mini-batch for weight update allows more efficient learning from the experience compared to updating the weight with respect to a single instance. Secondly, it reduces the variance of the updates as the successive updates are no longer correlated as would otherwise in the typical setting [29].

4.2.3.5 Objective Function

As mentioned previously, our DNN model aims to approximate optimal q-value. This is done by adjusting a set of weights specified in the model. Likewise most machine learning problems, the so-called "learning" is to optimize an objective function in an iterative fashion. This leads to defining the objective function to optimize. One of the central ideas behind RL is to estimate the state-action function in an iterative fashion according to the Bellman optimality equation [28] defined as the following:

$$\begin{aligned} q_*(s,a) &= \mathbb{E}[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') | S_t = s, A_t = a] \\ &= \mathbb{E}_{s'}[r + \gamma \max_{a'} q_*(s', a')] \end{aligned}$$

Where $q_*(s, a)$ is the optimal state-action value function and s' is next state after taking action a. Using the same idea, the value iteration algorithm solving the Bellman would then be:

$$q_{t+1}(s, a) = \mathbb{E}_{s'}[r + \gamma \max_{a'} q_t(s', a')]$$

Thus the state-action value function $q(s, a)$ can be estimated by minimizing a function of temporal difference (TD) error. Here we have decided to use Huber Loss on the ground that it is less sensitive to outliers. Huber loss is defined as:

$$l(x, y) = \begin{cases} 0.5(x - y)^2, \text{ if } |x - y| < 1 \\ |x - y| - 0.5, \text{ otherwise} \end{cases}$$

And the cost function is defined as

$$\text{target} := y = r + \gamma \max_{a'} q(s', a', w)$$

$$\mathcal{L}(w) = \mathbb{E}[l(y, q(s, a, w))]$$

To calculate the cost function, after each action is taken, we store the event tuple into replay memory, D. Then we sampled mini-batch of size n from the D (denoted as $(s_i, a_i, r_i, s'_i), i = 1 \dots n$). Thus the loss calculated at time t is :

$$\mathcal{L}(w_t) = \frac{1}{n} \sum_{i=1}^{n} l(r_i + \gamma \max_{a'_i} q(s'_i, a'_i, w_t), q(s_i, a_i, w_t))$$

However, the target q-values, $y = r_i + \gamma \max_{a'_i} q(s'_i, a'_i, w_t)$ depend on the same weight w as our q-values $q(s_i, a_i, w_t)$. Our estimated q-values move closer to the target q-values at each iteration. And the target q-values would move toward the same direction. Therefore, it introduces instability in the training and could possibly lead to oscillations. [32] (i.e. as we move closer to the target, the target moves further away). To overcome this problem, we introduced another network called *Target Network*. It serves as a clone of the original network with initial weights set fixed as the original network's weight. Then we update the weights to the original network's weight for every certain number of moves, C.

We parameterized the target network with weights $\bar{w}$, and the cost function become

$$\mathcal{L}(w_t) = \frac{1}{n} \sum_{i=1}^{n} l(r_i + \gamma \max_{a'_i} q(s'_i, a'_i, \bar{w}), q(s_i, a_i, w_t))$$

In which we periodically update fixed parameters $\bar{w} \leftarrow w_t$ for every C iterations, here C we set equal to 100.

4.3 Experiments

We did several experiments ranging the velocity of the bat. The velocity of a bat is defined as the distance travel per move, and the distance is specified by the length of the vector that is added to change bat's position. This length ranges from 1 to 5 units in each move setting the

time at single unit. This hypothetically represents different bat species. It is known that, each species of bat flies at different velocity. The size of the bat is kept fixed for simplicity, although this is customizable.

Variable	Values
Bat size	20 cm square object
Bat Velocity	1-5
Training site size	$60X60m^2$
Training without obstacle	No obstacle
Training with obstacle	Random shape obstacles

Table 3: Experiment setup

After setting the bat size and velocity we need to fix the training environment. We train the bat first in an environment where there is no obstacle. The reasoning behind this is not to take any unnecessary turns while there is no obstacle to reach a goal. Afterwards, we train the bat creating random obstacle of different shapes in a $60X60m^2$ environment. We did not use the actual forest to train the model as the 2-D representation of the forest is too sparse to train a bat. However, after proper training, we evaluate the model in provided forests.

4.4 Dataset

After the training phase, the bats have been tested on a set of unseen forest structures to evaluate the performance of taking actions according to the policy it learned. During this evaluation process, we extract a set of features based on the result of each action taken. Those included

- a_t : action taken at time t.
- r : reward received after the action is taken at t.
- Distance to obstacle for a specific action(θ) where $\theta = -20 \ldots 20$: Distance to the obstacle in a given direction specified by the angle (θ)

These features will be used for evaluating the performance of the model and support few claims.

	experiment	time	speed	gamma	signal1	signal2	signal3	distance_to_goal	action	orientation	reward
0	1	0	5	0.9	0.060	0.0450	0.0975	390.000321	19	0.999587	-1.0
1	1	1	5	0.9	0.050	0.0400	0.0500	390.189513	14	-0.894853	-1.0
2	1	2	5	0.9	0.050	0.0250	0.0600	390.356974	14	-0.816312	-1.0
3	1	3	5	0.9	0.035	0.0300	0.0450	390.493279	14	-0.738623	-1.0
4	1	4	5	0.9	0.025	0.0375	0.0375	390.590199	19	-0.660965	-1.0

Figure 22: Synthetic Bat movement dataset

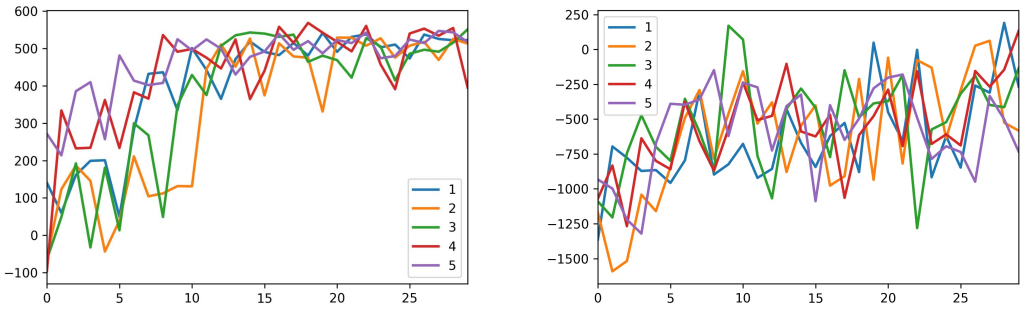
4.5 Evaluation

The performance of reinforcement learning algorithm is evaluated based on the policy it learns and how much reward it gets taking actions. During the learning phase, as the agent faces more random states, the expectation of getting higher rewards increases during deployment. Therefore, if there is sufficient learning before any agent is deployed in real-world, the policy it learns would bring better result and take proper actions in respective situations. However, if any agent has to learn policy during deployment, it may never get to a stage where it can take action which maximizes the expected reward. In this section, the objective is to evaluate the movement dataset and analyze the behavior of going across the forest.

To evaluate the performance of bats movement, we defined a metric called *Success Rate*. It is defined as, the percentage of number of time a bat travels across forest without hitting a tree.

4.5.1 Cumulative Reward trend

To access the training progress, it is customary to plot the cumulative reward of each episode against the number of episodes. The rationale behind this is that, had the training been setup correctly, the cumulative reward to increase and finally reach a stable range as the number of training episodes increases. The 23a shows the cumulative rewards trend under different bat velocity when there is no obstacle. Figure 23b shows the trend of cumulative reward under different velocity trained with obstacles. For both of the cases, we can see an upward slope of the trend. It is worth to note that, even for less than 50 episodes, the model achieved stable accumulated reward due to efficient reward scheme. It is also important to note that in each episode the agent takes 2000 moves, in the case when the agent reaches the goal, another goal will be randomly initiated on the other side of the forest.



(a) Cumulative rewards without obstacle

(b) Cumulative rewards with obstacle

Figure 23: Cumulative rewards over training episodes

In the figure 23b, we can see almost similar trend for training the bat with obstacles. It is worth noting that, in the environment with obstacles drastically increase the chance of having negative reward for each move. Consequently, we have mostly negative values as the cumulative reward in y-axis. In addition, there seems to be an increase in volatility in cumulative reward between each episode.

4.5.2 Relationship between success rate and velocity

When comparing the success rate with different velocity, we can observe an interesting trend in the figure 24. At different sites we can see that, faster flying bats seem to have higher success rates in almost every sites. However, the relationship between bat velocity and the ability of avoid obstacle needs further analysis and more rigorous explanations.

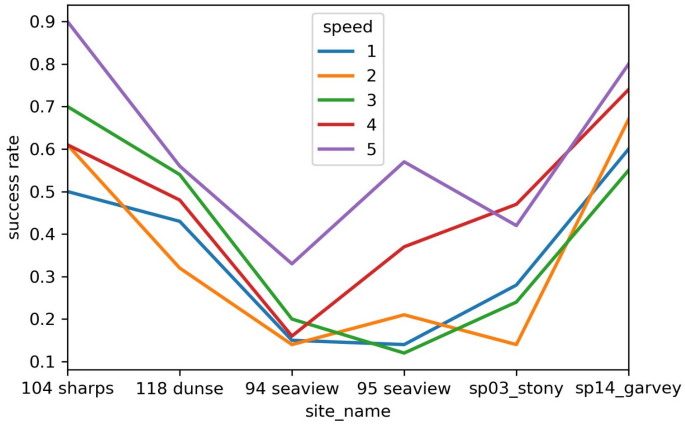
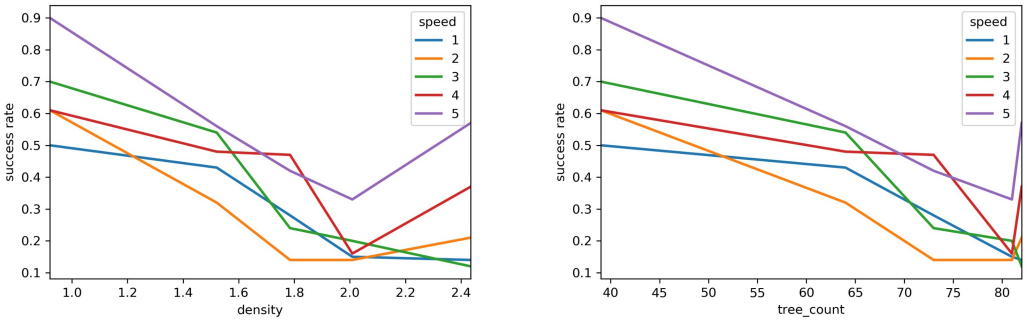


Figure 24: Success rate at different sites

4.5.3 Relationship between Success Rate and Forest Density

Intuitively, the success rate of going across the forest should differ depending on the density of obstacles in the forest. In this section, we have analyzed success rate at different sites. When comparing the Success rate vs Forest Density, we can see that there is a downward trend in the graph 25a. As the forest density increases, the success rate is gradually decreasing. This is reasonable since high density means there will be more trees leading to higher chance to hit a tree. This is true for bats with all different velocities.



(a) Success rate vs Forest Density

(b) Success rate vs number of trees

Figure 25: Success Rate of going across forest

This trend holds true even we compare the success rate with different number of trees. The figure 25b shows the relationship between them which has very similar trend.

4.5.4 Characterising the behavior of first hitting to Tree

The goal of this Reinforcement Learning algorithm is to reach the goal keeping maximum distance between obstacles. However, some actions might lead to hit a tree at a given circumstance. Therefore, in this section we analyzed the behavior of hitting a tree for bats with different velocity. In the figure 26, we can see that for all the forest sites in x-axis, faster flying bats hits the tree earlier than a slower moving bat moving. When the velocity of a bat is higher it takes fewer moves to hit the tree. This is a reasonable outcome, when facing an obstacle a faster-moving bat has fewer number of moves that it can take before getting to close the obstacle. In contrast, a slower-moving bat has more reaction time to decide.

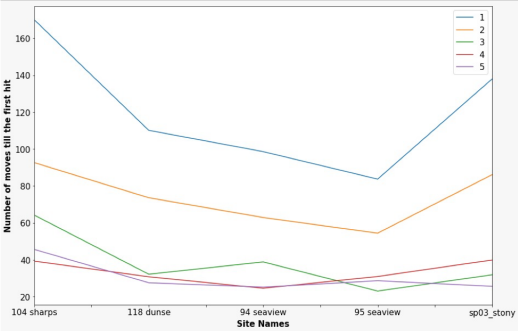


Figure 26: First Hit to Tree at different sites for different speeds

4.5.5 Collision Avoidance

Collision avoidance is one of the most important survival behavior of animals. The proposed work is based on avoiding collision with trees while simulating bats movement. In this section, we will analyze whether the bat chooses the appropriate action to avoid collision. Figure 27 shows the choice of action that a bat has made for different velocities and how far the obstacle is during that action. Here the Y-axis representing the distance to the obstacle from the bat for all actions. We can observe that bats are choosing the action where the distance to the obstacle is farthest. They usually find a path that is almost free of obstacles or move in the direction that is farthest from them. This analysis shows some promising results in avoiding obstacles.

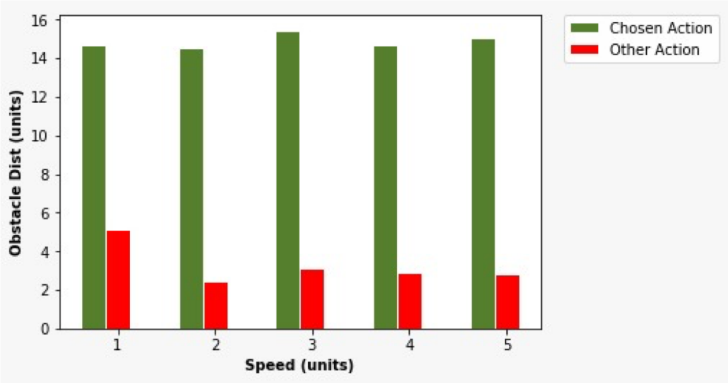


Figure 27: Obstacle Avoidance evidence

4.5.6 Relationship between Turning Angle and Velocity

In this section, we will analyze the relationship between the turning Angle of the bat with its velocity. Figure 28, shows the relationship between them at two different sites. It is easy to observe that, with the increase of velocity, the average turning angle also increases at a given site. This is reasonable as bats usually take a sharp turn in order to avoid the obstacles while moving faster. On the other hand, the slower-moving bats have more time to react, this suggests the reason to have smaller average turning angle.

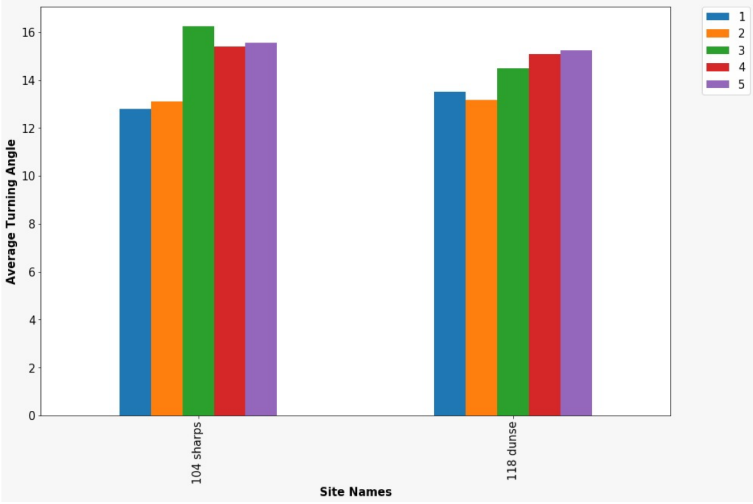


Figure 28: Relationship between Turning Angle and Velocity

4.6 2-D Simulation Software

Using Python and Kivy, an open source library equipped with novel user interfaces we have developed a simulation software of 2-Dimensional movement of Bats' around a given forest. Figure 29 is a snapshot of the movement in a given forest site. Here, white object represents the agent bat and the white dots are the trailing path of movement. Yellow objects are forest obstacles extracted from the cross-section of tree's from $5m$ to $8m$. Green dot represents the current destination goal for the bat. This software takes forest site and bat parameters as an input to simulate the movement. And it is easily customizable to learn turning angle for different parameters of bat using Deep Q Learning algorithm.

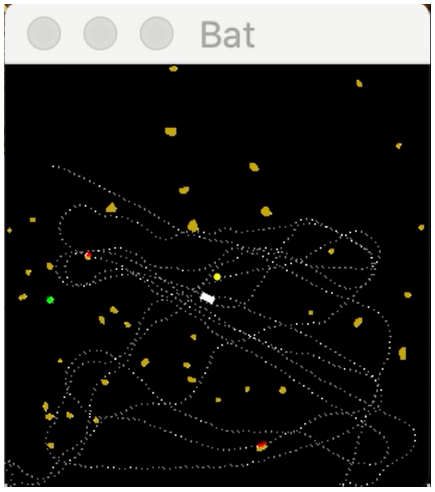


Figure 29: 2-D Simulation of movement

5 Discussion

The work has many significant sub-components that contribute equal importance in aggregating the final result. There are few improvement scopes for both of the phases, which can overall improve the final outcome in each problem.

In the first phase, the project faced different types of challenges from aspect of evaluation, algorithm selection with corresponding parameter tuning, and forest inventory parameters selection. One of the key requirements of the project is to find individual trees from the LiDAR data of each forest site. Due to the absence of GPS information in the LiDAR dataset, it was hard to verify the estimated tree position with the original one. For the estimation of the number of trees, we have applied both raster and cross-section based algorithm to overcome the absence of actual data. There are discrepancies between the tree count of both of the methods due to parameter tuning (which depends on the density of trees in the forest). To overcome this, the parameters should be tuned automatically based on forest data, which would make the analysis process more efficient and easy to interpret.

For the second phase, we made some rather heavy assumptions to simulate bats movement around the forest to simplify the original problem. We have assumed that the same species of bat has fixed size, moves with a fixed velocity, and flies at a certain height without any wingspan movement. In addition, while moving around the forest, we abstract away any impact regarding the environmental input. In reality the environment has a substantial impact on wild animals' habitat and ecological behavior. Seasonal, temporal, weather, foraging ecology has tremendous influences on movement patterns and behavior, which remained out of the scope for this work. Finally, to talk about the movement of bat precisely, the characteristics of movement patterns differ depending on the intention of movement such as foraging, migration, reproduction which has not been considered while defining the problem scope.

6 Future Work

The project is a key initial step to simulate animal movement around the forest in an unsupervised fashion. The way the problem has been defined is unique and can be useful in many studies of ecological behavior. However, there is a need for a deeper analysis of algorithms used in forest inventory analysis and find better approaches addressing the limitations in each step. There are few directions where more works can make the results more promising and bring a significant impact on both forest inventory analysis and animal movement. Due to resource limitation, the study of finding the best performance and its result in each of the proposed solutions were left behind.

The following ideas could bring dynamic dimension in the project:

- There is an utmost importance of evaluating forest inventory analysis using real survey data of different forest site with detailed tree information, which involves different tree parameters.

- Additional canopy parameters such as canopy cover, density and gap along with leaf area index and leaf are density can be introduced to obtain a comprehensive Forest inventory analysis. Details can be found in Appendix A.

- For bat movement simulation, getting rid of the erroneous assumptions sequentially will make the movement more realistic and applicable to the real ecological study. Most of the species of bat have a sonar sensory navigation system. Adding these properties in the observed behavior of bat will make the movement pattern more comparable to the studies already existing in the literature. To mimic the echolocation behavior, we used three sensors to sense the surrounding obstacle density. This made the task easier than the actual bat's sensing behavior because bat analyzes each echo separately and processes to detect the environment. So, there is a scope of improving sensing ability.

- There is a need for more rigorous training to learn accurate flying parameters using the proposed Deep Q Learning algorithm. This will help to study and compare the observed behavior of real bats and mimic the behavior in virtual world, which can be useful in developing artificial robot bats.

- Hypothesis testing is indeed crucial at the next stage to assess the significance of the claims that have been made analyzing the synthetic bat movement data. This will ensure the claims are not causal due to the randomness of data.

- Most of the ecological movement studies in the literature involve the collection of real-world animal movement data. In the future, a controlled experiment or tracking bat in the forest can bring enormous data to compare the movement of proposed artificial bat movement data with the real-world one and verify flying parameters.

- The last but not the least is the three-dimensional extension of the movement simulation. A major and realistic extension of this work would be to find the flying parameters in 3-D and visualize in a real forest. In addition, the proposed obstacle avoidance algorithm is a straightforward and better approach is using control theory for 3-D rotations.

7 Conclusion

The use of LiDAR technology undoubtedly has excellent potential in forest ecosystem research. There has been a significant amount of work on processing LiDAR data; only a few of the work are customizable and emphasized Terrestrial data. The developed tool in this work bridged the gap between different approaches and contributed to filling it by identifying individual trees with the spatial information and characteristics in 3-Dimension. It also allows users to take detailed advantageous step by step analysis with intermediate outputs in a user-friendly online Shiny application hosted in a cloud environment. Currently, it can provide information such as tree positions, diameter, and heights given a LiDAR dataset.

The characterization of forest objects is followed by the task of simulating movement patterns of the animal Bat going across the forest. In this study, we managed to simulate and

investigate bats movement behavior in a controlled environment under a set of assumptions and the adoption of Deep Q Learning algorithm. Our proposed algorithm learned to take appropriate action (turning angle in this case) avoiding forest obstacles. More efforts have been given to make the model generalize for simulating the movement pattern of other animals. Although in some aspects, the simulation does not represent the real-world behavior and much more simplistic than a real bat. It is worth to note that, this project is only the first step to formally defining the problem of bats movement simulation in an unsupervised fashion, throwing few questions to be answered by the research community.

8 Contribution

The primary objective of this project is to model bats movement around forest habitat. Under this umbrella, there has been a massive amount of work of efficient processing of the given LiDAR forest dataset. At the initial stage of this work, the main focus remained limited to analyze Aerial and Terrestrial LiDAR dataset of different sites. As a result, this project delivered a forest inventory analysis tool as a byproduct, which has been hosted on the Melbourne Research Cloud. The main contribution of the work is that our client has already started using it and, even more, requested some extensions, which involves finding more forest and tree specific parameters. Currently, this tool is focused on terrestrial LiDAR data. It provides essential information including but not limited to tree count, canopy height, object detection, and tree diameter, implementing different algorithms to evaluate and cross-validate the result of inventory analysis.

The second phase of the project is focused on Modelling the bats' movement around the forest. The primary target of this part of the project is to learn to take proper action avoiding collision with trees using Deep Q Learning algorithm. To the best of our knowledge, this is the first work that has adopted Deep Q Learning to learn Bat flying parameters in two dimensional world. It is worth to mention that there is no dataset available to model the movement of bats around the forest. The only available resource is the LiDAR data for the entire project, and simulating the movement is indeed an open-ended problem. A big portion of our time was devoted to doing the literature review for the sake of defining the problem of Bat's movement with a proper methodology to simulate it.

This project involves two mutually exclusive vast sub-problems which is hard to finish on one academic year. It was hard for a team of four people to formulate the problem and ensure best result trying different algorithms at the same time. In our point of view, we should divide the scope of the project and assign two different teams for each of the tasks. This can bring more promising and meaningful results.

References

[1] World Bank, "World development indicators," 2019, [Online; accessed 13-November-2019]. [Online]. Available: https://data.worldbank.org/indicator/AG.LND.FRST.ZS

[2] World Wildlife Fund, "World wildlife fund," 2019, [Online; accessed 16-November-2019]. [Online]. Available: https://wwf.panda.org/our_work/forests/importance_forests/

[3] V. Tewari, "Forest inventory, assessment, and monitoring, and long-term forest observational studies, with special reference to india," *Forest science and technology*, vol. 12, no. 1, pp. 24–32, 2016.

[4] J. Trochta, M. Krůček, T. Vrška, and K. Král, "3d forest: An application for descriptions of three-dimensional forest structures using terrestrial lidar," *Plos One*, vol. 12, no. 5, 2017.

[5] C. Cabo, C. Ordóñez, C. A. López-Sánchez, and J. Armesto, "Automatic dendrometry: Tree detection, tree height and diameter estimation using terrestrial laser scanning," *International journal of applied earth observation and geoinformation*, vol. 69, pp. 164–174, 2018.

[6] N. Brodu and D. Lague, "3d terrestrial lidar data classification of complex natural scenes using a multi-scale dimensionality criterion: Applications in geomorphology," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 68, pp. 121–134, 2012.

[7] T. de Conto, K. Olofsson, E. B. Görgens, L. C. E. Rodriguez, and G. Almeida, "Performance of stem denoising and stem modelling algorithms on single tree point clouds from terrestrial laser scanning," *Computers and Electronics in Agriculture*, vol. 143, pp. 165–176, 2017.

[8] M. Isenburg, "Lastools-efficient tools for lidar processing," *Available at: http: http://www. cs. unc. edu/˜ isenburg/lastools/[Accessed October 9, 2012]*, 2012.

[9] Q. D. Team *et al.*, "Qgis geographic information system," *Open Source Geospatial Foundation Project*, 2016.

[10] W. Li, Q. Guo, M. K. Jakubowski, and M. Kelly, "A new method for segmenting individual trees from the lidar point cloud," *Photogrammetric Engineering & Remote Sensing*, vol. 78, no. 1, pp. 75–84, 2012.

[11] M. Dalponte and D. A. Coomes, "Tree-centric mapping of forest carbon density from airborne laser scanning and hyperspectral data," *Methods in ecology and evolution*, vol. 7, no. 10, pp. 1236–1245, 2016.

[12] C. A. Silva, A. T. Hudak, L. A. Vierling, E. L. Loudermilk, J. J. O'Brien, J. K. Hiers, S. B. Jack, C. Gonzalez-Benecke, H. Lee, M. J. Falkowski *et al.*, "Imputation of individual longleaf pine (pinus palustris mill.) tree attributes from field and lidar data," *Canadian journal of remote sensing*, vol. 42, no. 5, pp. 554–573, 2016.

[13] Melbourne Research Cloud, "Melbourne research cloud," 2019, [Online; accessed 14-November-2019]. [Online]. Available: https://docs.cloud.unimelb.edu.au/

[14] Department of Agriculture and Water Resources, "Australian forest profile," 2019, [Online; accessed 14-November-2019]. [Online]. Available: https://www.agriculture.gov.au/abares/forestsaustralia/profiles

[15] A. Haywood, A. Mellor, and C. Stone, "A strategic forest inventory for public land in victoria, australia," *Forest Ecology and Management*, vol. 367, pp. 86–96, 2016.

[16] G. Liu, J. Wang, P. Dong, Y. Chen, and Z. Liu, "Estimating individual tree height and diameter at breast height (dbh) from terrestrial laser scanning (tls) data at plot level," *Forests*, vol. 9, no. 7, p. 398, 2018.

[17] R. Bullen and N. McKenzie, "Bat airframe design: flight performance, stability and control in relation to foraging ecology," *Australian Journal of Zoology*, vol. 49, no. 3, pp. 235–261, 2001.

[18] ——, "Bat wing airfoil and planform structures relating to aerodynamic cleanliness," *Australian Journal of Zoology*, vol. 55, no. 4, pp. 237–247, 2007.

[19] J. Müller, M. Mehr, C. Bässler, M. B. Fenton, T. Hothorn, H. Pretzsch, H.-J. Klemmt, and R. Brandl, "Aggregative response in bats: prey abundance versus habitat," *Oecologia*, vol. 169, no. 3, pp. 673–684, 2012.

[20] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature inspired cooperative strategies for optimization (NICSO 2010)*. Springer, 2010, pp. 65–74.

[21] R. Eberhart and J. Kennedy, "Particle swarm optimization," in *Proceedings of the IEEE international conference on neural networks*, vol. 4. Citeseer, 1995, pp. 1942–1948.

[22] Y. Li, "Deep reinforcement learning," *arXiv preprint arXiv:1810.06339*, 2018.

[23] I. Eliakim, Z. Cohen, G. Kosa, and Y. Yovel, "A fully autonomous terrestrial bat-like acoustic robot," *PLoS computational biology*, vol. 14, no. 9, p. e1006406, 2018.

[24] G. Wang, L. Guo, H. Duan, L. Liu, and H. Wang, "A bat algorithm with mutation for ucav path planning," *The Scientific World Journal*, vol. 2012, 2012.

[25] I. K. Ibraheem, F. H. Ajeil, and Z. H. Khan, "Path planning of an autonomous mobile robot in a dynamic environment using modified bat swarm optimization," *arXiv preprint arXiv:1807.05352*, 2018.

[26] U. Goel, S. Varshney, A. Jain, S. Maheshwari, and A. Shukla, "Three dimensional path planning for uavs in dynamic environment using glow-worm swarm optimization," *Procedia computer science*, vol. 133, pp. 230–239, 2018.

[27] T. Hirakawa, T. Yamashita, T. Tamaki, H. Fujiyoshi, Y. Umezu, I. Takeuchi, S. Matsumoto, and K. Yoda, "Can ai predict animal movements? filling gaps in animal trajectories using inverse reinforcement learning," *Ecosphere*, vol. 9, no. 10, 2018.

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[29] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[30] O. Marom and B. Rosman, "Belief reward shaping in reinforcement learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[31] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[32] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[33] C. Hopkinson and L. Chasmer, "Testing lidar models of fractional cover across multiple forest ecozones," *Remote Sensing of Environment*, vol. 113, no. 1, pp. 275–288, 2009.

[34] M. Vehmas, P. Packalén, M. Maltamo, and K. Eerikäinen, "Using airborne laser scanning data for detecting canopy gaps and their understory type in mature boreal forest," *Annals of Forest Science*, vol. 68, no. 4, pp. 825–835, 2011.

[35] J. Roussel and D. Auty, "lidr: Airborne lidar data manipulation and visualization for forestry applications. r package version 1.0. 0," 2018.

[36] R. Meng, J. Wu, F. Zhao, B. D. Cook, R. P. Hanavan, and S. P. Serbin, "Measuring short-term post-fire forest recovery across a burn severity gradient in a mixed pine-oak forest using multi-sensor remote sensing techniques," *Remote sensing of environment*, vol. 210, pp. 282–296, 2018.

[37] C. M. Listopad, R. E. Masters, J. Drake, J. Weishampel, and C. Branquinho, "Structural diversity indices based on airborne lidar as ecological indicators for managing highly dynamic landscapes," *Ecological Indicators*, vol. 57, pp. 268–279, 2015.

[38] H. Fang, W. Liu, W. Li, and S. Wei, "Estimation of the directional and whole apparent clumping index (aci) from indirect optical measurements," *ISPRS journal of photogrammetry and remote sensing*, vol. 144, pp. 1–13, 2018.

A Future Work Extension

Potential variable	Description	Resource
Canopy Cover	Estimated as the number of first returns above 7.5 m height (i.e. top height of the under-storey) divided by the number of all first returns; Higher CC indicates higher cover projected within the crown boundary on the horizontal plane. Default the cutoff height to 7.5, back make the height adjustable.	[33]
Leaf Area Index	A measurement of leaf area at the plot level that indicates the leaf cover over the ground.	GitHub
Canopy Gap	Gap is one of the main metrics that can describe the effect of fire in terms of forest cover deterioration (gap = 100-cover) [34]	GitHub
Leaf Area Density - mean	Mean leaf-area density (LAD) of the vertical profile of each 1-m bin between 7.5 m and the total tree height; values close to 1 indicate denser crown biomass [35]	GitHub
Canopy Density	Estimated from the number of all lidar points above 7.5 m height divided by the number of all returns; a higher CD represents a higher density of plant material within the crown boundary above 7.5 m; crown density can be used as a measure of the amount of leaf material available for maintaining tree productivity and vigour [36]	Package
Height strata	Describes the vertical structure of the canopy at different height bins which indicates that in which strata higher biomass found.	Package
Evenness index	A modification of the Shannon–Weiner diversity index	[37]
Clumping Index	Calculated as effective Leaf Area Index	[38]

Table 4: Potential future variables for forest sites

Stand structure/LiDAR metrics	Description
Percentile height (percentiles 5 to 95 in 5% increments)	Indicates that the metric represents the i percentile height value of the points in the cell above the height cutoff . Height cutoff to be able to set throughout
Mode canopy height	Indicates that the metric represents the mode height value of points in the cell above the height cutoff i.e. indirectly represents whether the average trees are larger or smaller
Mean canopy height	As per percentile heights
As per percentile heights	As per percentile heights
Standard deviation of height	As per percentile heights
Mean quadratic height (QAV)	Mean of the quadratic height, square root of the arithmetic mean of squared values
Some others	Canopy Relief Ratio, Rumple index, Bicentile [36], Skewnewss, Kurtosis, Canopy reflection sum, LADmax [35], LADmin [35], LADmaxperHt [35], LADminperHt, LADcv, LADsd

Table 5: Potential future LiDAR specific parameters