Requirements Specification

December 3, 2021

Team LumberHack

Sponsor:
Dr. Andrew J. Sánchez Meador

Mentor:
Melissa D. Rose

Team Members:
Matthew Flanders
Jenna Pedro
Thomas Whitney
Colin Wood

Version 1.0

Client Signature:_____

Table of Contents

Introduction 3

Problem Statement 3

Solution Vision 4

Project Requirements 7

- Functional Requirements 7
- Non-Functional Requirements 13
- Environmental Requirements 14

Potential Risks 15

Project Plan 17

Conclusion 19

Works Cited 21

Introduction

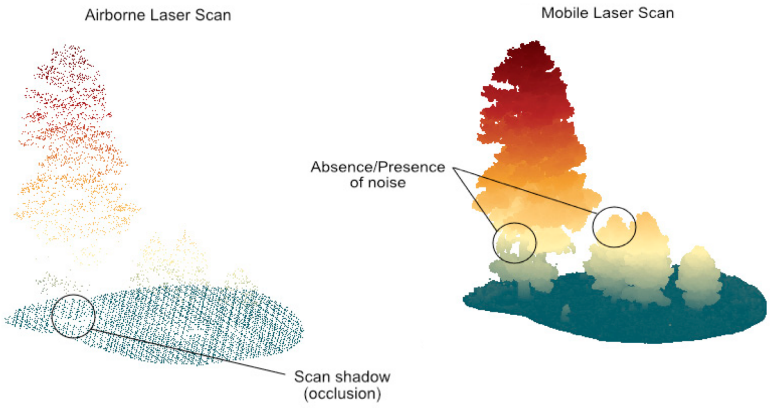
Forest ecosystem health is at the center of many large-scale environmental problems that the world faces today. Over the last century, forest management policies have heavily focused on timber production and fire exclusion. The focus on production has led to high tree densities and more combustible material in forests. This has resulted in a greatly increased risk of catastrophic wildfires, droughts lasting longer and happening more frequently, as well as increased recruitment of invasive species taking over forests. With climate change advancing, forest ecosystems are facing greater threats than ever before. Scientists and forest managers work on restoration projects to try and bring forests back to a healthy state. These projects include forest surveying, thinning, and cleaning of high danger, dense areas. The scale of these projects ranges from a few acres of land to many square miles of forests. With better surveying tools and analysis, restoration efforts are quicker and yield more useful information. The sponsor for this project is Dr. Andrew J. Sánchez-Meador, an Associate Professor at the School of Forestry, as well as the Executive Director at the Ecological Restoration Institute at Northern Arizona University. Dr. Sánchez-Meador's research focuses on using quantitative forest ecology for multi-scale forest ecology and restoration projects. His recent research applies practical interpretations of complex data sets to analyze individual tree growth and provide new ways of visualizing data for science communication. By using remote sensing technologies such as light detection and ranging, or lidar, Dr. Sánchez-Meador and other ecologists are able to collect and interpret large complex data-sets to model individual tree growth and indicators of forest health.

Ponderosa Pine Forest
Flagstaff, AZ

Problem Statement

Due to forests covering large spatial areas over difficult terrain, collecting observations for an individual tree can be challenging. Lidar instead serves as a way to collect

informative-data for individual trees over large forested areas. Lidar is a remote sensing method that uses light pulses combined with GPS data to create precise, three-dimensional point clouds replicating the shape and characteristics of trees and their surrounding environment. Airborne lidar scanning (ALS) has been the traditional method for surveying forests, utilizing a lidar sensor attached to the bottom of an airplane or helicopter. ALS covers a larger range than other forms of lidar, such as mobile lidar, but the number of data points is low (around 10 points per square meter). With mobile lidar scanning (MLS), data is instead taken from the perspective of the forest floor. Mobile lidar can have a couple thousand points per square meter which results in denser, higher-resolution point clouds.



Current tools are lacking in a few ways such as:

- No focus on mobile lidar
- Steep and complex learning curves
- Usually programming knowledge is required
- Visualization can be improved upon
- No graphical user interfaces (GUI's)

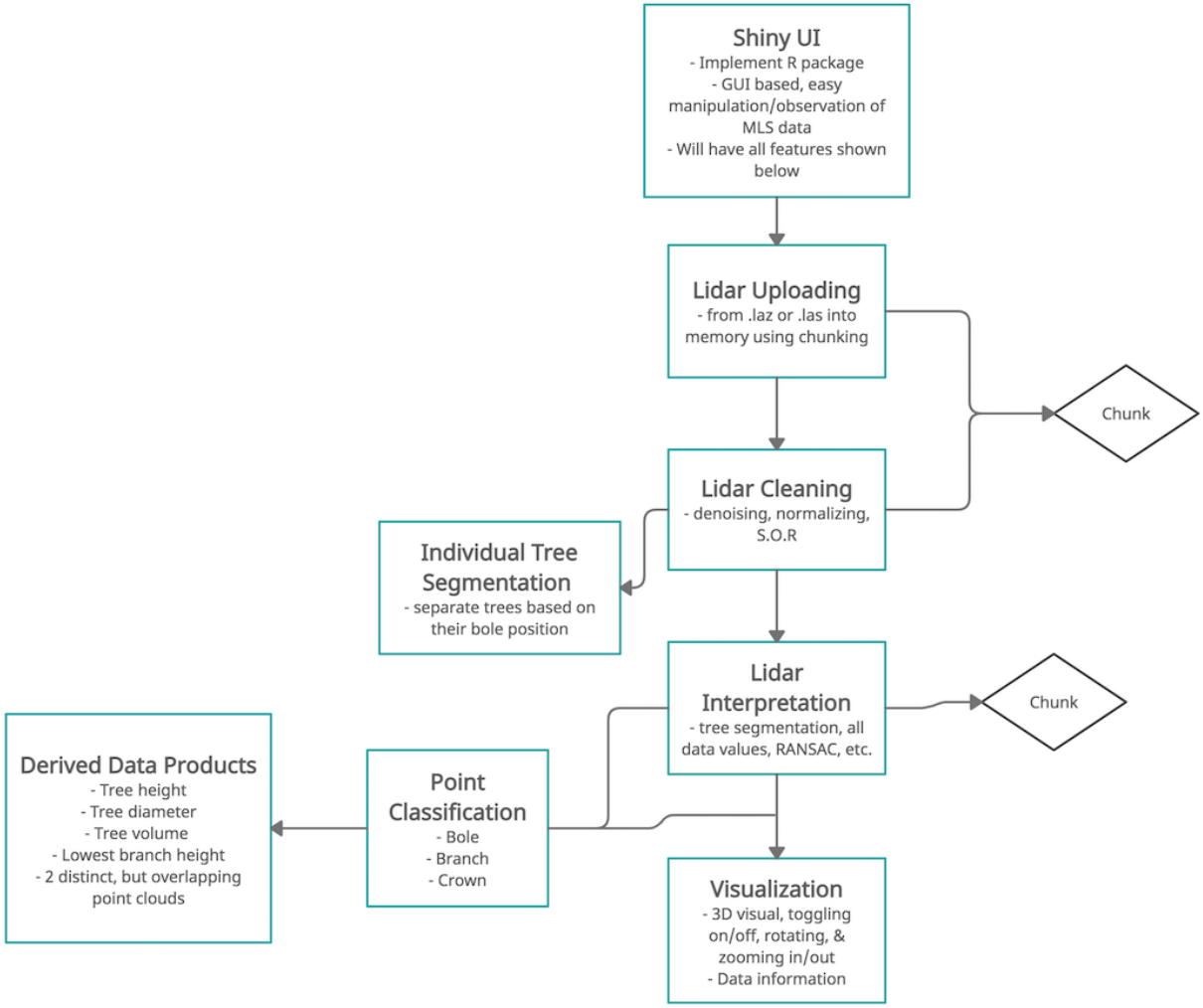
With MLS gaining popularity, researchers such as Dr. Sánchez-Meador need a tool that is built to effectively evaluate MLS data and will yield informative ecological characteristics.

Solution Vision

The envisioned solution of this project will serve as a resource for researchers and ecologists to analyze their own MLS data. As part of the final product, the team plans to create an easy-to-use application with the following features:

- Data Upload
    - Users will be able to upload only .laz and .las files.
    - Users will be able to upload as many files as they would like, under 10GB.
- Data Cleaning and Denoising
    - Once data is uploaded,it will be cleaned and normalized using a statistical outlier removal algorithm (SOR).
- Individual Tree Segmentation
    - After the uploaded data has been automatically cleaned, the user can run a tree segmentation algorithm to separate trees based on their bole position.
- Point Classification
    - Individual points in the point cloud will be classified based on their position on a tree such as bole, branch, or crown.
- Derived Data Products
    - Users will then be able to select specific trees and obtain relevant measurements such as tree height, tree diameter, and lowest branch height.
    - Users will also be able to see how two temporally distinct but overlapping point clouds match up
- Data Summaries/Visualization
    - The GUI will also display statistics in both a 3D interactive point cloud, as well as on a table.

The final product will be a single portable tool that any ecologist can access and use. With a straightforward GUI, the team plans for users to be more focused on their research than on learning a new piece of software. The project will focus on creating a MLS specific application to support the growing number of researchers utilizing MLS. The features explained above will be wrapped into a Shiny R application that will be portable and accessible from the web.



Solution Vision Workflow

Project Requirements

This project's requirements will be laid out below to ensure that the team develops a tool with the features outlined in the previous section. The requirements will be laid out into three sections;

- Functional
- Non-Functional
- Environmental.

Functional Requirements

Shiny UI

The final application will implement the R package Shiny for its user interface. Shiny is a package that provides a web framework for building applications using the R language. The application will be GUI based and allow for easy manipulation and observation of MLS data. As a web based app, it will be portable and accessible on most machines. Functions that are implemented will require minimum programming knowledge to use and will be mostly based on GUI features such as buttons, menus, and 3d point cloud visualization.

Example Shiny Web App

The application will provide a way to allow users to upload their own MLS data. The data will then be pre-processed and cleaned using functions that are built into the lidR package that Dr. Sanchez-Meador has helped develop. The GUI will include a way to allow users to modify RANSAC parameters for cylinder fitting and parameters for point classification. Users will be able to visualize the data in both a 3D point cloud, as well as on a table.

The web app will be an all in one tool that abstracts away the technicalities of traditional lidar point analysis. The application will include tools and functions that provide a basis for ecological research projects through its Shiny based GUI.

Data Upload

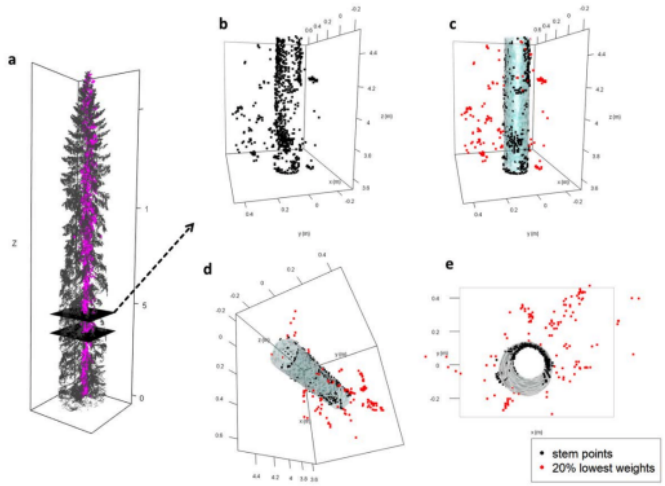
The first thing users will do is upload their own data to the project. Both .laz and .las file types will be able to be uploaded. The data will be uploaded through a GUI using the Shiny package. Multiple file uploads will be supported up to a 10GB total. When multiple files are uploaded, users will be able to select which file they would like to work on in the application.

Data Cleaning and Normalizing

Once user data has been uploaded, it will be automatically cleaned. The cleaning process will use the lidR package's built in function *classify_noise* to remove outliers using SOR methods. Once this function is run, the data is clean. Next, the *normalize_height* function, also in lidR, will be run to normalize the dataset with the ground at 0. This will allow for all data to be processed similarly.

Individual Tree Segmentation

In order to derive data products like the tree diameter, the team will be using cylinder shape fitting with the tree center size at 1.37m from 1.17m to 1.57m and 0.4m to 0.5m segment. This will use RANSAC (Random Sample Consensus) for tree boles with attribute and error reporting. RANSAC extracts arbitrary random minimal sets from the point data. Even with the proportion of outliers present, it is still a reliable process. The process starts with taking a slice of a tree and separating trees. In order to get a good cylinder fit, there needs to be about 15 to 30 random points and this should be done about 10 to 20 times.



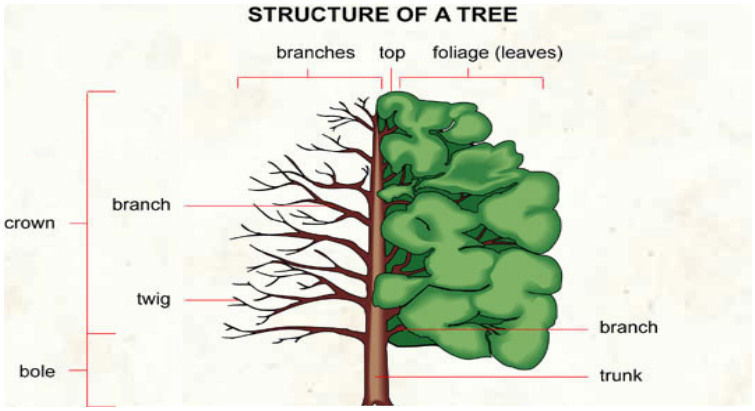
Using the Principal Component Analysis(PCA), cylinder fitting is used

to estimate above-ground biomass and tree location. First, the points will outline a 1.37m sample circle. Then the tree location is defined with the projection of the circle's center. For error reporting, about a 20% margin of error and 80% accuracy would be great for the team's software. Even if the error of what is given from a 1 inch tree compared to a 20 inch tree will be the same.

Point Classification

The application will include functions that allow for classification of points in the 3D point cloud such as bole, branch, and crown. These classifications will be used for grouping of similar points into their individual tree sections. The application will include point classifications such as the bole, branch, and crown of a tree, and others including logs that are laying horizontally on the ground.

The classification component is important for identifying composition of individual trees as well as the ability to view the point cloud with specific groupings. For example, low branches and lots of dead wood on the forest floor act as fuel for fires. Being able to classify and view just branches and dead trees on the ground would greatly improve the time it takes to clean up high fire danger materials.



Functions will be implemented in the backend of the application that identify and classify which points are which. This will be designed using algorithms that identify verticality and horizontality as well as grouping algorithms such as nearest neighbor to classify points into the groups mentioned above. The classifications will then be implemented into the GUI of the application to allow the user to select which set of points they want to view.

Identification of key characteristics within a point cloud

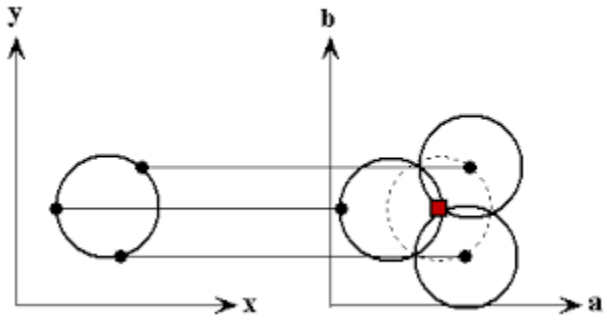
One of the key requirements necessary for the project is the ability to extract features from point cloud data, these features include tree location, diameter, and the identification of tree

structures such as branches and boles. These tree structures are identified using the arrangement of groups of points and describing them as more linear, planar, or spherical orientation, while tree diameter and height is identified by finding the center and radius of a circle that best fits a group of points. Both of these methods need to be done quickly by improving current methods used in lidR and CloudCompare which do not utilize multiple processors or do not have efficient memory calculations. Once these features have been derived from the point cloud they can then be used for the creation of the 3D representation of the point cloud.

In order to identify the key characteristics in the point cloud the software will perform linear algebra calculations to determine how groups of points are oriented, as well as feature extraction techniques to identify the center and radius of a circle that best fits the points.

Using linear algebra, eigenvalues will be calculated and saved to be used in the calculation of linearity, planarity and sphericity. These values can then aid in the identification of how these points are oriented. Once the orientation of points in a cloud have been determined they can then be used later in the classification of tree structures such as a tree branch or bole.

Another feature of interest is determining the best fit circle for a group of points, to do this a Hough transformation will be done. A Hough transform can be described as taking an iterative approach of an increasing radius and drawing a circle around a random sample of points using that radius. Using this collection of circles the program will identify a point that has the highest number of circle intersections and then test the fit of the candidate center and radius. Once a candidate center and radius is able to encapsulate 80% of points within a threshold distance of the circle's edge it will be used as the best fit circle for a group of points.

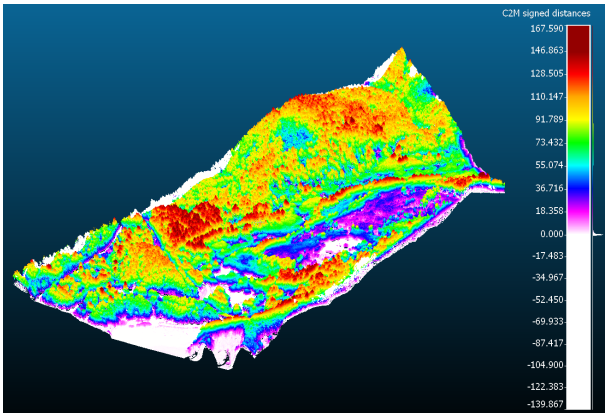


Hough transform to find center of a circle

These features can be time consuming to calculate due to the linear algebra needed for solving for matrix determinants to get eigenvalues and the iterative approach used in the Hough transform to find the center and radius of a circle. Here efficient use of previously calculated eigenvalues and multithreaded computation is needed to derive the key features needed for 3d visualization.

Change detection of an individual attribute over time

Another function that this application will include is having the ability to calculate differences between ecological attributes including height, volume, above-ground biomass, and individual tree location across two points in time. There are preprocessing steps on the MLS data before calculating differences in observed ecological pertinent attributes. The CloudCompare software could be used to preprocess these steps. First, the data would have to be denoised with a noise filter tool, then use a CSF(Cloth Simulation Filter) filter for ground classification, and lastly, manually classify the trees and segment them into individual trees.



Cloth Simulation Filter for ground classification

To calculate the tree height, the team will use the height-above-ground method. The height above ground is calculated by subtracting the height value of the highest point in the tree point cloud from the height of the ground surface. The team will use CloudCompare for the above-ground calculation using the cloud/mesh distance tool and compute the maximum height above the ground of each tree with MATLAB. The volume of the tree is computed by the stem curve. This is automatically derived from the point cloud, between the minimum height and height of the ground surface. A method to estimate above-ground biomass of large trees is by

three-dimensional tree modeling of the MLS point clouds. To get accurate measurements, cylinder shape fitting is needed.

Registration of two, temporally distinct but overlapping, point clouds

One of the important uses of the software is to be able to monitor changing forest conditions over time. To accomplish this two point clouds need to be loaded into the program, and then be properly aligned so that accurate comparisons can be made of the same plot of land. Due to difficulty in aligning point clouds precisely, the software will use point cloud specific tools that can fine tune the orientation of the point clouds and identify the best fit.

Stretch Goals

Desirable stretch goals fall into one of the following two categories: making the software more easily *usable* and making the software more easily *reusable*.

Descriptive Vignettes:

On the usability front, the team envisions small "vignettes" throughout the software, textual or graphic, that explain to the user how to best use/interact with a given feature or tool. These short bits of documentation need not be limited to simply explaining the user interface–they could also give insight into certain design software decisions that more advanced users, such as researchers, would find helpful and informative as they use the software. In this example, this would save the researcher from needing to go search through the official documentation for small but important pieces of information. Opportunities to create such vignettes will arise during development.

CRAN Certification:

On the reusability front, the team has one major stretch goal: to package the software as a CRAN package. CRAN, which stands for Comprehensive R Archive Network, is the official repository for third-party R packages. CRAN requires that hosted packages include documentation and error handling and meet testing requirements. While making the software CRAN-compliant will require additional work, it will make it more visible to the scientific

community, and thus more likely to be used. Aiming for acceptance into the CRAN repository will ensure a well-designed and fully functional piece of software at the end of the project.

Non-Functional Requirements

Shiny UI

The user interface (UI) will be clear and easy to use in its capabilities. This means that only necessary functionality will be included such as the ability to clean and process data to produce meaningful results in both tabular and 3 dimensional forms. It will be easy to use as descriptions and more information on specific functions will be provided to the user. The package will be portable and on the web allowing for anyone to access and use it. This will be tested by presenting peers with the application and observing navigation of someone who is not familiar with lidar analysis tools.

Has the ability to calculate differences between observed ecologically pertinent attributes

To clearly visualize the ecologically pertinent attributes of trees, the data will go through preprocessing steps. The data has to be denoised first with the noise filter tool in CloudCompare. Then the CSF filter will allow users to see points from the ground until the tops of the trees. Lastly, the team will manually classify the trees and segment them into individual trees. After the data is pre-processed, the calculated attributes of large trees will be visualized by three-dimensional tree modeling of the MLS point clouds. To get a clear display, cylinder shape fitting using RANSAC is needed.

Cylinder (or mesh or multiple sphere) shape fitting with tree size at 1.37m

RANSAC requires to get about 15 to 30 random points for a good cylinder fitting and needs to do this process 10 to 20 times. It is expected that it performs with high speed to return the cylinder. The client states that minutes are fine and ideal, but seconds would be golden. For error reporting, about a 20% margin of error and 80% accuracy would be great for the team's software.

Classification of specific points

Classifying of specific points allows for breaking down the point cloud to allow for viewing and analysis of specific sections. The project should minimally have the ability to classify points as ground or bole. The client has stated that 80% accuracy for classification is a good baseline to hit, and anything above that would be great. Classifications for branches and crown of a tree would be a nice inclusion but are not a minimum requirement for the project. The classification algorithm should take no more than 1-2 minutes as the eigenvalues allowing for classification will already be calculated and ready to use.

Registration of two, temporally distinct but overlapping, point clouds

There are two options that will be provided to users in order to fulfill this requirement. First, highly reflective markers can be used as a point of reference if both scans include them and have been placed in the same location. Second, point clouds can be manually rotated by a user within the GUI by clicking and dragging the point clouds into a rough estimation of being aligned.

Once the scans have been aligned a registration tool such as the CloudCompare Iterative Closest Point(ICP) algorithm can be used to finely tune the orientation of the two point clouds. After the ICP algorithm has completed comparisons can be made to determine what changes have been made in the given plot.

Environmental Requirements

R Programming Language

The most significant constraint imposed is that the team must use the R programming language and environment. There are multiple reasons for this. First, the R programming language sees widespread use in the scientific community, and is straightforward for non-programmers to understand and write. This latter aspect is important because it makes complex software more accessible. Furthermore, many contemporary software tools for lidar processing are written in R, including LidR, the main package from which the team plans to

borrow functionality. Second, the client is experienced with the R programming language, meaning he will be able to give us any necessary guidance.

GeoSLAM

One further environmental restriction is the GeoSLAM algorithm. GeoSLAM is proprietary software that processes the raw lidar output from the mobile lidar machine. In essence, it stitches together the many thousands of lidar snapshots captured by the machine into a single, three-dimensional lidar output file. GeoSLAM must be run as the very first step in the processing pipeline, otherwise the lidar output is unusable. No third-party alternative to GeoSLAM exists: every mobile lidar machine must be used with accompanying software provided by the manufacturer. This restricts our pipeline in that all input to our application must have already been run through GeoSLAM, or another proprietary tool if the mobile lidar data is from another manufacturer's mobile lidar scanner.

Potential Risks

The team anticipates four key areas of risk associated with the final project, some internal and some external.

Internal risks

Internal risks are those over which the team has direct control. It is up to the team to ensure that the likelihood of these risks is mitigated, and the team will be able to directly take action to undo any undesired outcomes if they are realized.

Poor code development and/or documentation

Poor code development and/or documentation are problematic for multiple reasons. Poor documentation might frustrate first-time or non-technical users and ultimately deter them from using our application. Unstructured and/or unclear code may deter collaborative development, and would make code maintenance more difficult, possibly causing the project to fall out of maintenance. Proper code development and documentation are important considerations for any project, but the team emphasizes them here because of the project’s special situation in the scientific and open source communities. There is a moderate likelihood of these issues arising,

and the team will be best suited to avoid them by following a single coding standard, and avoiding time crunches that may contribute to rushed, confusing code or forgotten documentation.

A difficult to use GUI

A graphical user interface that is inadequate in one or more ways is a second internal risk. Because the UI will be the face of the software package, it is crucial that it be intuitive, able to control all necessary aspects of the underlying software, and be non-buggy. An unintuitive or buggy UI will impede users, and if the shortcomings are great enough will cause them to abandon the application altogether. A UI that is not capable of performing all desired functional requirements will render those underlying software features unusable. Because the team plans to use Shiny to package the front-end UI, and because Shiny is a widely used and thoroughly developed library, there is a small likelihood that the UI becomes buggy and/or nonfunctional. Nonetheless, the team must take steps to ensure that UI features are laid out intuitively, and that all functionality is reachable via the UI.

External risks

External risks are those over which the team has no, or only limited, control. These may arise from data inputted by a user of the application, third-party software, or any other outside interactions with the team's software. Because the team will have limited ability to fix any such issues if they arise, it is important that they be mitigated effectively ahead of time.

Inaccurate computations by third-party software

Accurate output from the core processing tools of the application is crucial. Inaccurate results, at a minimum, would complicate use of the software and deter use. More seriously, inaccurate results that go unnoticed could be put into public databases, used as the basis of research grants, get published, and so on. Clearly, if the software does nothing else it must report accurate results. The team is specifically concerned about the accuracy of third-party computations because these are less likely to be thoroughly vetted than computations that the team develops itself. Instead of simply copy-pasting desired functionality from other sources and moving on, the team needs to closely evaluate any repurposed code. This should include an

assessment of the reliability of the source (e.g. Github statistics, code review), testing, and approval from the client. If these guidelines are followed, the likelihood of incorporating inaccurate or misbehaving software is lessened but still not zero.

Computations and/or visualizations that overwhelm Shiny

Shiny, the R framework that the team plans to employ in the browser, is primarily concerned with front-end development, that is, data presentation (and not data processing). Because one of the core goals of the project is to make the application non-specialist friendly, the team envisions using Shiny to interface with the user at all stages of the pipeline, including for the many heavy-duty data processing steps. Thus, much back-and-forth between Shiny and the user's operating system will be necessary to run processes, access input and write output, allocate memory, and so on. Although initial investigation of Shiny's functionality suggests that these tasks should be achievable, they are not Shiny's strong suit. Care will need to be taken to make sure that these interfaces are not only functional but also efficient–enough time is already going to be used performing the core processing.

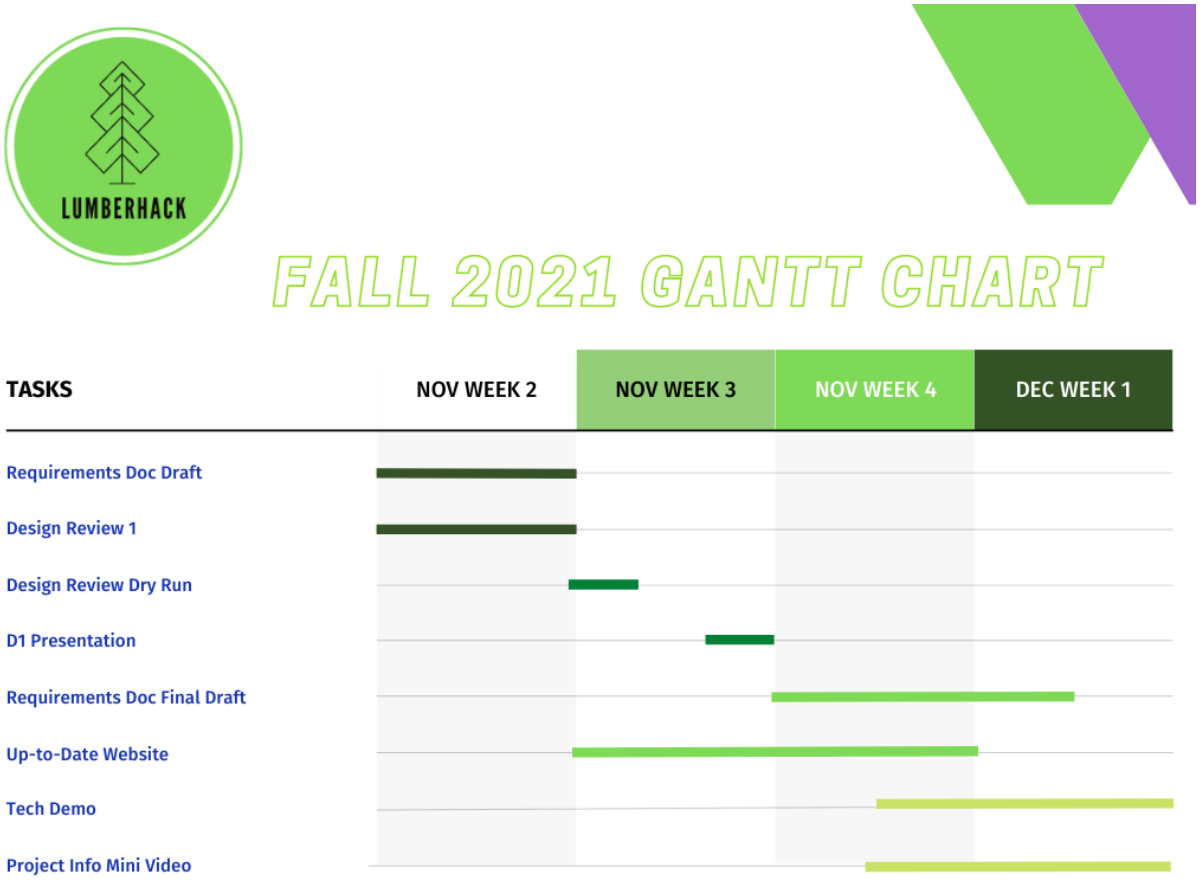
Additionally, the team envisions some sort of three-dimensional and possibly interactive visualization of the lidar data at the end of the pipeline, which will be run inside of Shiny. Depending on how Shiny handles view updating with Javascript inside of the browser, this may become difficult or impossible. The team will need to investigate this issue as we become more familiar with Shiny and its underlying power. Dialing back the complexity or the interactivity of the visualization may become necessary. There is a moderate likelihood that one of these two problems materializes.

Project Plan

As of right now, eight milestones have been identified during project development. The major points of the team's plan is to calculate the decomposed eigen and hough transformation values, classify differences between observed individual tree attributes, use RANSAC for cylinder shape fitting, classify points, register two temporally distinct but overlapping point clouds, and use Shiny for user interface. The team's stretch goals are to incorporate vignettes to

explain the functionality of the package and the Shiny appPackage uploaded and passing all CRAN checks so it can be an official “CRAN” package.

In order to complete these milestones, each has been expected date of completion in the months to comeBelow will first show the Gantt chart of the team’s plan this semester.



Once the team heads into the spring semester, there will be more specific major and minor milestones added onto the project plan. Below is a rough breakdown of the team’s plan next semester, presented in a Gantt chart.

Conclusion

Through the experience gathered by Dr. Sánchez-Meador as the executive director of the Ecological Restoration Institute, it has been found that there is a lack of efficiency in the current methods used to monitor forest health. Traditional methods typically involve groups of three individuals manually measuring trees, recording data, and then later uploading collected data to a computer for processing. This method is time consuming, labor intensive, does not provide a complete picture of forest structure, and the results are difficult to interpret. More modern approaches that use lidar are implemented for the purposes of ALS monitoring where a top down view of the forest is provided and not a ground level view. These modern approaches to forest monitoring also require technical knowledge that can be challenging for non-technical individuals to use, or the software is poorly optimized for processing data.

This project intends to solve these problems by providing a more efficient method of forest monitoring. First it will require less people for data collection by using mobile LiDAR to create a point cloud of the forest. From the point cloud the project will then provide an easy to use interface via a Shiny App for the upload, cleaning, interpretation, and visualization of lidar data. From the functionality of the app users will be provided with a tool that can effectively derive and display pertinent ecological data, as well as see ground level views of the forest structure all within an easy navigable web app.

So far requirements have been gathered over meetings with the sponsor and technological feasibility has been conducted on possible implementation methods. With a clear understanding of the project needs and possible tools to address them, the team looks forward to implementing a solution to these challenges.

Works Cited

Dietrich, James. “Filtering Lidar Data by Height above Ground.” *Filtering LiDAR Data by Height above Ground*, 1 Jan. 1970, http://adv-geo-research.blogspot.com/2017/10/filtering-lidar-data-by-height-above.html.

Donager JJ, Sánchez Meador AJ, Blackburn RC. Adjudicating Perspectives on Forest Structure: How Do Airborne, Terrestrial, and Mobile Lidar-Derived Estimates Compare? *Remote Sensing*. 2021; 13(12):2297. https://doi.org/10.3390/rs13122297

Heo, Han Kyul, et al. “Estimating the Heights and Diameters at Breast Height of Trees in an Urban Park and along a Street Using Mobile Lidar.” *Landscape and Ecological Engineering*, Springer Japan, 10 Apr. 2019, https://link.springer.com/article/10.1007/s11355-019-00379-6.

Herrero-Huerta, Mónica, et al. “Automatic Tree Parameter Extraction by a Mobile Lidar System in an Urban Context.” *PLOS ONE*, Public Library of Science, https://journals.plos.org/plosone/article?id=10.1371%2Fjournal.pone.0196004.