

# **User Manual**

May 5, 2022



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**

## Table of Contents

Introduction	2
Installation	3
Configuration and Daily Operation	3
Maintenance	4
Troubleshooting	5
Conclusion	6

## Introduction

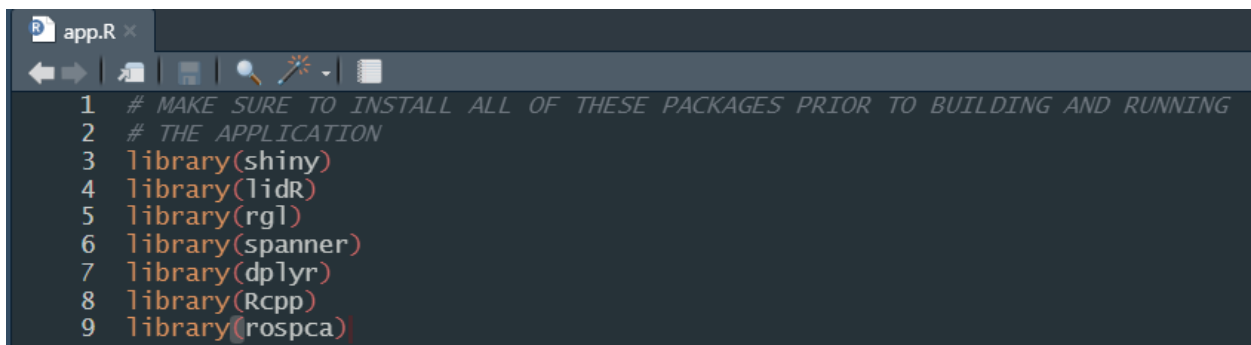
We are excited to provide detail into how to use Team LumberHack's mobile lidar scan (MLS) processing application. The MLS processing application is an all inclusive tool that has been designed and developed to meet the needs of forestry researchers and other mobile lidar users. Some of the key features include:

- Uploading of multiple las/laz files to the application
- Cleaning and normalization of lidar scans with the click of a button
- Pre-processed 3d point cloud view of a lidar scan
- Ransac circle fitting to calculate useful tree characteristics
- Viewing of output in both a 3d point cloud and 2d table

The purpose of this user manual is to help you, the client, successfully install, administer, and maintain this tool for your research needs going forward. The team's goal is to make sure that you are able to use the software that we have built for years to come.

## Installation

As part of the installation, our team will assist in installing the software on your own machine. The application is hosted on the github repository found here: [https://github.com/tommywhitney/mobile\\_lidar](https://github.com/tommywhitney/mobile_lidar). To download the application, you must clone the repository from the github page. Next, you will need to extract the entire folder to a location of your preference. Once extracted, you can launch the project by opening the *mobileLidar.Rproj* file in RStudio. Navigate to `/R/app.R` inside of the `mobile_lidar` directory. Open `app.R` and install all of the packages that are loaded at the top of the file.



```

1 # MAKE SURE TO INSTALL ALL OF THESE PACKAGES PRIOR TO BUILDING AND RUNNING
2 # THE APPLICATION
3 library(shiny)
4 library(lidR)
5 library(rgl)
6 library(spanner)
7 library(dplyr)
8 library(Rcpp)
9 library(rospca)

```

### *Current package dependencies*

Next, ensure that you have the latest version of R devtools installed. More information can be found here: <https://www.r-project.org/nosvn/pandoc/devtools.html>. With all of the package dependencies and devtools installed, you are now ready to move onto the configuration and daily operation steps to build and run the application.

## Configuration and Daily Operation

To use the mobile lidar application, complete the following steps:

- 1) Open RStudio
- 2) Navigate to the directory where *mobileLidar.Rproj* is saved
- 3) Set RStudio to build documentation using Roxygen by clicking the *Build* tab, then *More -> Configure Build Tools* and click the box for generating the appropriate documentation.
- 4) Select OK and then build the appropriate documentation file by clicking the *Build* tab, then *More -> Document*.

- 5) Load the package and restart your R session, again using the *Build* tab.
- 6) In the console, either press “ctrl + shift + I” or type `devtools::load_all(".")`
- 7) Also in the console, either type `runapp('app.R')` or open `app.R` in RStudio editor and click on *Run App* button at the top right of the *File Panel*.
- 8) After a few moments, you will see a pop-up window displaying the Shiny web application.
- 9) To upload a file, click on *Browse...* button
- 10) Then, select any las/laz file that doesn't exceed 10GB from the *Select File* drop down menu
- 11) To clean the data and click on the *Clean Data* button. A pop-up message window will be shown for the cleaning process.
- 12) On the *Table View* tab, you will be able to see the laz/las file cleaned and its metadata.
- 13) To open a 3D point cloud view prior to tree segmentation, click on the *Draw Point Cloud* button and view it on the *Point Cloud View* tab.
- 14) To run RANSAC, click on the *Run Ransac* button.
- 15) To view segmented trees and data products on a 2D table, click on the *Draw Slice* and *Table* button. When that process is completed, select the *Slice View* tab.

## Maintenance

Maintaining the project will involve a couple separate tasks. First, and probably most important, will be managing R package dependencies. The R dependencies are too many to list here, but during the installation process all required packages will be enumerated by R. Alternatively, use the built-in “`available.packages()`” function for a package of interest to see its dependencies. Noteworthy here are the Shiny package, which drives the front-end, and the `lidR` package, which powers much of the LiDAR processing pipeline and has many dependencies itself. At the time of writing, there were no dependencies that required a less-than-current (historical) release. However, in the event that some future version of any dependency is breaking, an older release for that dependency might have to be installed, or the cause of the issue addressed.

Second, auxiliary files that accumulate in the project's directories may have to be cleared occasionally for space reasons. These will most likely be .csv files in the src/ and root directories that are used to pass LiDAR data between R and C++ functions. Removing these at any time other than in the middle of processing will be harmless. Other temporary files such as .out and .temp files created by the C++ compiler can also safely be removed at any time other than runtime.

Access to the github repository will be arranged with Dr. Sánchez-Meador during the final project delivery meeting on May 5, 2022. If some problem arises, email Thomas Whitney, the owner of the repository, at [tdw226@nau.edu](mailto:tdw226@nau.edu).

## Troubleshooting

During installation there is a chance that dependency problems will arise. Based on the team's experience, the problems are operating-system dependent, with Mac being substantially more problematic than Windows. None of the team members installed the project on Linux. It is difficult to predict exactly which problems will arise, because each person will have different versions of operating systems, different package managers, and different libraries installed. Unfortunately, R is known for its frustrating package management system. There are however tools that can make this process easier. Rstudio comes with package management tools, and Packrat is another alternative for managing R packages. R package dependency issues are not the only that can arise. These rely in turn on various third party libraries that may or may not have to be installed manually. On the Mac operating system, using the Homebrew package manager was helpful in this regard.

It is difficult to predict which other problems might arise in the future, but an outline of the likely cause for a variety of problem types is given here:

- Any issue with the user interface (e.g. non-responsive buttons, web page is blank, files do not upload) is likely to be an issue with Shiny. All shiny code is located in the R/app.R file. Run the app interactively in an R interpreter and look for errors.
- If after file upload, the cleaning normalization steps fail to complete, the problem likely lies in the lidR package. Check to see if updates to the lidR package have been issued.

Run the program interactively in an R interpreter and see at which step execution fails—the lidR package’s cleaning and normalization functions have helpful outputs at each step.

- If the number of trees identified does not match the expected/known number of trees in the scan, or if multiple trees are being identified as a single tree, the issue likely lies with the DBSCAN (density-based spatial clustering of applications with noise) algorithm, which segments trees from one another. There are parameters to this algorithm that may need to be changed, such as the minimum number of points needed to classify a point as an inlier, and the radius used for point detection. Very dense forests may pose a need to adjust these parameters, for instance. Read about DBSCAN here for more information: <https://en.wikipedia.org/wiki/DBSCAN>.
- If tree measurements (e.g. radius, location, lean angle) are significantly inaccurate, the problem likely lies with the RANSAC (random sample consensus) algorithms used to fit shapes to the point cloud data. There are several parameters to this algorithm that can be adjusted, but an explanation of these is outside the scope of this document. The parameters have been set to a good medium between extremes, as far as the team can tell. Point cloud quality may also be an issue here. Use a point cloud visualization technology such as CloudCompare to inspect the trees. Scans of tree trunks should be obviously cylindrical, with only a reasonable amount of outliers, and with a decent amount of point density.

## Conclusion

In conclusion the CS capstone team has been very excited and has worked hard to develop the mobile lidar application and we hope that it has productive use in real world applications in the improvement of forest ecosystem health. While we all look forward to our next steps after graduation we are happy to answer any questions that may arise during the use of this product.

Best wishes:

Colin Wood, [cvw29@nau.edu](mailto:cvw29@nau.edu)

Jenna Pedro, [jtp275@nau.edu](mailto:jtp275@nau.edu)

Matthew Flanders, [mtf83@nau.edu](mailto:mtf83@nau.edu)

Thomas Whitney, [tdw222@nau.edu](mailto:tdw222@nau.edu)