Forest Canopy Detection from Lidar Point Cloud and Orthophoto

Biswadeep Sen and Kuldeep Meel

Abstract

Light Detection and Ranging (Lidar) has been widely applied to characterize the structure of forests as it can generate 3D point data with high spatial resolution and accuracy. Forest canopy detection, usually derived from the canopy height model, can be used in a wide variety of problems in forestry and ecology. While there has been a lot of progress in automated tree crown segmentation, height based detection of tree canopy or clusters of trees close to each other from various data sources has not been explored. In this study, we do a comparative study of different types of algorithms for accurately delineating tree canopy based on height from a Lidar point cloud and orthophoto. We use a density based clustering approach, an object detection based approach on the orthophoto and finally some individual tree crown segmentation based approaches.

1 Introduction

The forest canopy is a structurally complex and ecologically important subsystem of the forest. Forest canopy cover is defined as the proportion of the forest floor covered by the vertical projection of the tree crowns (Jennings, Brown, and Sheil 1999). Researchers use the information related to detecting forest canopy in a wide variety of tasks. Firstly, forest canopy cover is a multipurpose ecological indicator frequently used for distinguishing different plant and animal habitats. Many researchers use canopy cover to assess forest floor microclimate and light conditions and estimate the leaf area index (LAI). The LAI quantifies the photosynthesizing leaf area per unit ground area (Lowman and Rinker 2004). Secondly, accurate detection of tree canopy enables estimation of the timber volume of a forest (Maltamo et al. 2004). Thirdly, canopy cover detection can help ascertain the burn severity of a forest and hence help in fire management, and post-fire vegetation recovery (Yin et al. 2020). Finally, some researchers have used it for studying the relationship between tree canopy cover and land use in urban settings(Jim 1989; Rowntree 1984).

Traditionally, tasks related to the systematic collection of data about forest information were field-based (Hyyppa et al. 2001), but the advent of remote sensing technology has created new avenues for reliable estimates of forest attributes in large areas. Remote sensing is the science of obtaining information about objects or areas from a distance, typically from aircraft or satellites. There has been a heightened interest in the usage of remote sensing methods in forest management activities in the past decade due to the availability of new datasets and significant technological advancement (Saremi et al. 2014; Hudak, Evans, and Stuart Smith 2009; Hansen et al. 2015). The usage of these methods are widely expected to complement field measurements, and it is mainly because field measurements take a long time and have a significantly higher cost (Silva et al. 2016).

While there is substantial literature on the individual tree crown segmentation and tree detection from overhead imagery, our work is the first of its kind to detect the forest canopy as a whole based on height in Indonesian peatlands. We choose indonesian peatlands because these are some of the least understood peatlands globally and are important because of their function as carbon storage (Page, Rieley, and Banks 2011). Also Indonesia is a low resource country and hence does not have a lot of field instruments. So automated methods for detecting forest canopy cover can very very useful for monitoring these forests. In this work, we propose three different automated methods for detecting and segmenting forest canopy from various data sources. Firstly, we try a density-based clustering approach on the orthophoto. Secondly, we try a deep learning-based object detection on the orthophoto, and finally, we try an individual tree crown segmentation based approach on the Lidar point cloud. We perform an extensive empirical evaluation of these methods with the ground truth. Our best performing method gets an intersection over union score of 0.901 with respect to the ground truth.

The rest of the paper is structured as follows: Section 2 discusses some essential terms related to the paper. In section 3, we talk about the prior work done in this field. In section 4, we talk about the data used and the challenges. In section 5, we discuss the details of our approaches and the challenges encountered. Subsequently, we provide details of the evaluation in Section 6 and conclude in Section 7.

2 Related work

While our work provides methods for height-based forest canopy segmentation, related work by Deschne et al. (2016) provides a 3-step method for forest stand delineation according to tree species. The method relies on the computation of lidar and multispectral features at different levels, used for supervised tree species classification. They test their approach in a study site located in a mountainous forest in the East of France. The final results obtain a good matching with the segments delineated by human annotators (Dechesne et al. 2016).

Saleh et al. calculate the accuracy of individual tree crown segmentation algorithms in detecting and segmenting individual trees in a research site located in Waringin Timur Regency, Central Kalimantan Province in Indonesia. The compare the accuracy of the different methods by comparing with field data (Saleh et al. 2020). In contrast, our work provides methods for detecting and segmenting forest canopy as a whole, not just individual trees.

Apart from this, the work by Neuville, Bates, and Jonard explores the usage of the HDBSCAN clustering algorithm for accurate segmentation of tree stems in a deciduous forest(Neuville, Bates, and Jonard 2021). Sackov, Kulla and Bucha perform a comparative study between two tree detection algorithms for the estimation of forest stand and ecological variables from airborne LiDAR Data in Central European Forests(Sačkov, Kulla, and Bucha 2019). Korhonen et al. provide a comparison between different field-based techniques for estimation of the forest canopy to determine the best method for large scale forest inventory. (Korhonen et al. 2006)

Apart from this the work by Neuville, Bates, and Jonard explores the usage of the HDBSCAN clustering algorithm in accurate segmentation of tree stems in a deciduous forest(Neuville, Bates, and Jonard 2021). Sackov, Kulla and Bucha perform a comparative study between two tree detection algorithms for the estimation of forest stand and ecological variables from Airborne LiDAR Data in Central European Forests(Sačkov, Kulla, and Bucha 2019). Korhonen et al. provides a comparison between different field based techniques for estimation of forest canopy to determine the best method for large scale forest inventory (Korhonen et al. 2006).

3 Background

DBSCAN is commonly used to detect connected components in a dataset (Ester et al. 1996). Height based detection of forest canopies also makes us do a similar task to detect groups of similar trees in a height range. One of the other advantages of DBSCAN is that it does not assume the number of clusters in the dataset a priori. Our dataset also does not know the number of clusters beforehand, which makes DBSCAN an excellent algorithm to use.In this algorithm, all points in the dataset are divided into three groups: core points, (density-) reachable points and outliers. There are two main parameters: minPts and epsilon. A point p is a core point if at least minPts points are within distance epsilon of it (including p). All points within a distance of epsilon from p are said to be directly reachable from p. A point q is directly reachable from p if there is a path from p to q such that each pair of adjacent points is at a distance of at most epsilon from each other. All points which are not reachable from any other point are called outliers or noise points. During execution, the algorithm finds the points in the epsilon neighbourhood of every point and identifies the core points with more than minPts neighbours. Then it finds all the connected components of the core points while ignoring the non-core points. Then it assigns each non-core point to a cluster if it is in an epsilon neighbourhood of that point. Otherwise, it is assigned to noise.

Tree detection from overhead imagery is a significant problem in ecosystem research, and a lot of scientific and commercial research relies on the accurate delineation of tree crowns. Deep learning based object detection techniques has been used in academic literature to detect the individual tree crowns. One of the main advantages of deep learning over methods that use handcrafted pixel features is that convolutional neural networks (CNNs) directly delineate objects of interest from the training data, improving transferability among projects. CNNs learn hierarchical combinations of image features that focus on object-level representations of objects rather than pixel-level. Also, neural networks are re-trainable to incorporate the idiosyncrasies of individual datasets, allowing us to refine models with new data without discarding old data. (Weinstein et al. 2019). The DeepForest model (Weinstein et al. 2020) is such a pretrained model that can be used to detect treecrowns from overhead imagery. This model was trained using a semi-supervised approach using data from the National Ecological Observatory Network (NEON) site at California's San Joaquin Experimental Range. The site contains woodland consisting of a variety of trees, including live oak (Quercus agrifolia), blue oak (Quercus douglasii), and foothill pine (Pinus sabiniana) forest. The authors also had access to Lidar point cloud data for this region for training purposes. They used unsupervised tree detection algorithms to create approaches to create the initial self-supervised tree predictions in the LIDAR point cloud. A bounding box was automatically drawn over the entire set of points assigned to each tree to create the training data. The authors used this dataset to train the deep learning model, which was later retrained on a small number of hand-annotated images to correct errors from the unsupervised detection. For this model, they used the retinanet one-stage detector. The way retinanet differs from other object detection frameworks is that it combines detection and classification into a single network, allowing faster training and decreasing the sensitivity to the number of box proposals (Lin et al. 2017). Here a resnet-50 backbone pre-trained on ImageNet was used as a backbone to retinanet(He et al. 2016).

4 Data

The data was collected from a 2,000 ha study site to build an extensive geospatial dataset. The researchers have organized five aeroplane flights with Lidar on this site every six months since 2015. The data collected from these flights consist of a LiDAR point cloud dataset and an orthophoto acquired dur-

ing each of the five flights. The point cloud obtained was used to construct a LiDAR canopy height model(CHM). The Canopy Height Model (CHM) denotes the height or residual distance between the ground and the top of objects above the ground. This includes the actual heights of trees, buildings, and other things on the earth's surface. Each point in the data contains the following components: x-coorodinate and y-coordinate of the point denoting its location, the normalized vegetation height, vegetation classification and Intensity.

4.1 Challenges:

One drawback we faced when we used the algorithms proposed was the presence of clouds in the point cloud. The pulses reflected by the clouds before they reached the ground give an abnormally high height value at that particular point. The field experts knew that all the trees in this region had a height of less than 50 meters. So we discarded any treetop that had a height less than 50 meters and then used the algorithm to calculate the individual tree crown segmentation.

5 Approaches and challenges

We deploy three different types of approaches on the datasets. For the first approach we do a density based clustering approach on the LiDAR point cloud. Then, we do an object detection based approach on the orthophoto and finally we do an individual tree crown based segmentation approach on the point cloud.

5.1 Density based Clustering approach

We define a new approach named canopyDetect for detecting forest canopy cover from Lidar point cloud. It takes four parameters: the maximum and minimum heights of the trees to cluster, a parameter ϵ and another parameter minPts. Once we have the maximum and minimum height, 'canopy detector' takes the x and y coordinates of all the points whose height lies within that range from the Lidar point cloud. Then it runs the DBSCAN algorithm with parameters ϵ and minimum points = minPts. Once we have the clusters, we construct convex hulls around them to form the canopies of that height range.

Algorithm 1 RQ

Input: Points,distFunc,ϵ,Q
Output: All neighbors of the pt Q

N = {}
for P in Points **do**
  if distFunc(Q, p) $\leq \epsilon$ **then**
    N $\cup$ P
return N

The main challenge is finding the appropriate values of the hyperparameters empirically. After trying out with different parameters we got the best results from minPts = 20 and $\epsilon = 0.9$.

Decreasing the value of ϵ resulted in creation of numerous small disjoint clusters. On the other hand increasing the

Algorithm 2 DBSCAN

Input: Points,distFunc,ϵ,minPts
Output: Clusters of Points

C = 0
for p in points **do**
  if label(p) $\neq$ None **then**
    continue
  $N \leftarrow$ RQ(Points, distFunc, p, ϵ)
  if |N| $<$ minPts **then**
    label(p) = NOISE
    continue
  C = C + 1
  label(p) = C
  S = N.delete(p)
  for q in S **do**
    if label(q) == NOISE **then**
      label(q) = C
    if label(q) $\neq$ None **then**
      continue
    label(q) = C
    N = RQ(DB, distFunc, q, eps)
    if |N| $\geq$ minPts **then**
      S $\cup$ N

Algorithm 3 canopyDetect

Input: L,ϵ,minPts,hmin,hmax
Output: Polygons denoting the tree crowns

L′ = {}
for i in L **do**
  if i.height $\leq$ hmax and i.height $\geq$ hmin **then**
    L′.append(i)
C $\leftarrow$ DBSCAN(L′, ϵ, minPts)
for i in C **do**
  i = convexHull(i)
return C

value of ϵ resulted in creation of a few big clusters which is also not appropriate for our work.

In case of the minPts parameter also we had similar results. Decreasing minPts created few large clusters and increasing it created many small clusters neither of which was useful for our purpose.

Figure 1: Output of canopyDetect on our dataset

5.2 Object detection based approach:

We can look at the canopy cover as just a collection of individual trees. So we can use deep learning-based object detection to detect the individual tree crowns from the orthophoto and then we can combine trees that have the height within our predefined height range to form the canopy. Even though RGB photos are less expensive to acquire than Lidar point clouds, they lack spatial information (Weinstein et al. 2019). So in this method, we combine the results from object detection with information from the Lidar point cloud to detect canopies in a particular height range. However, Deep Learning methods, including object detection, require massive amounts of data for training which we lack. Usually, thousands of labelled images are used to train the neural network, but in our case, we only have five images for each of the five flights at six-month intervals. So our best option is to use the pretrained DeepForest model on overhead forest imagery.

So we define canopyDetect2 in the following way: We use the pre-trained retinanet model DeepForest on the orthophoto and find the bounding boxes denoting the location of trees obtained by object detection. As previously mentioned, object detection on the orthophoto lacks information about the height of the trees. We need information about the height to form the tree canopies based on heights. So for this purpose, we used the findTrees function in the LidR package written in R (Roussel et al. 2020). This function uses a moving window approach to find the treetops in the point cloud (Popescu and Wynne 2004). The function allows an h_min parameter, which denotes a pixel's minimum height to be considered a treetop.

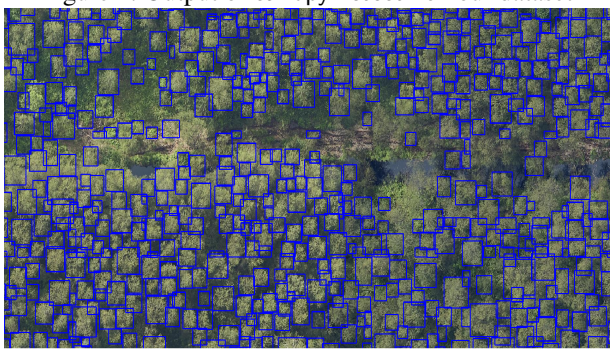
Once we get all the bounding boxes as an output to the Deepforest model and the treetops, we find all the boxes containing a treetop in the predefined height range. Then we take the union of all these bounding boxes to form the canopy using the cascaded_union function in the shapely package in python (Gillies 2013).

Algorithm 4 canopyDetect2

Input: L,Orthophoto,hmin,hmax
Output: Polygons denoting the tree crowns

```
BoundingBoxes = DeepForest(Orthophoto)
treetops = findTrees(L)
filteredTreetops = []
for i in treetops do
    if i.height ≤ hmax and i.height ≥ hmin then
        filteredTreetops.append(i)
finalBoxes = []
for i in BoundingBoxes do
    for j in filteredTreetops do
        if j in i then
            finalBoxes.append(i)
            break
return cascadedUnion(finalBoxes)
```

Figure 2: Output of canopyDetect2 on our dataset



5.3 Individual tree crown segmentation based approach

Individual tree crown segmentation is the process of individually delineating trees after spatially locating them and extracting height information (Roussel et al. 2020). We use an individual tree crown segmentation algorithm on the Canopy height Model(CHM) for this approach. This algorithm returns each segment of the image that contains a tree in the form of a polygon. Once we get this tree crown segmentation, we take the union of the polygons obtained to form the final canopy cover.

For the individual tree crown segmentation, we use an approach by Dalponte and Coomes(2016). This approach was originally used in a study area in the Italian Alps(Pellizano, Trento), which had an altitude of 900 meters to 2000 meters (Dalponte and Coomes 2016).

This approach goes through a collection of steps. First, a low pass filter smooths over the rasterized canopy height model to reduce the number of local maxima. Then the local maxima are located using a moving window approach. We can choose the moving window to be square shaped or circular. A pixel in such a moving window is designated as a local maximum if its value is greater than all other values

in the window and greater than some threshold height above the ground. Each local maximum is a region around which we construct a tree crown. We use the find_trees function from LidR package to find the local maximum (Popescu and Wynne 2004). Next, we extract the height of the four neighbouring points from the canopy height model and add them to the tree crown region if their vertical distance from the local maximum is less than some user-defined percentage of the local maximum height and less than some user-defined maximum difference. We iteratively keep repeating the procedure for all the neighbours of the cells now included in the tree crown region until no points can be further added. After obtaining the points for each tree crown, we create a convex hull around these points, and the resulting regions become the final tree crown segmentations (Dalponte and Coomes 2016).

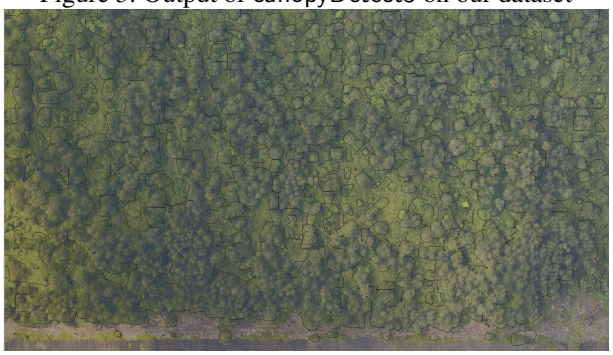
Once we obtain the tree crowns, we take the union of all these polygons to form the tree canopy. To create the union cascadedUnion function was used from the package shapely in Python.

Algorithm 5 canopyDetect3

Input: L,hmin,hmax
Output: Polygons denoting the tree crowns

```
treetops = findTrees(L)
filteredTreetops = []
for i in treetops do
    if i.height ≤ hmax and i.height ≥ hmin then
        filteredTreetops.append(i)
treeCrowns = DalponteCoomes(L, filteredTreetops)
return cascadedUnion(treeCrowns)
```

Figure 3: Output of canopyDetect3 on our dataset



6 Evaluation setup and Results

We conducted the training and evaluation on high performance machines on compute clusters with the following specifications: Intel Xeon Silver 4108 with Nvidia Titan V and Intel Xeon Silver 4108 with Nvidia Tesla V100.

We test and validate our methods on the fourth flight organised by INTPREP. Our goal is to segment the forest canopy consisting of trees above 15 meters. To evaluate the accuracy of our models, we use human-annotated labels as

Algorithm 6 Individual Tree Crown Segmentation (Dalponte and Coomes)

Input: L,treetops,hmin,hmax
Output: Polygons denoting the tree crowns

```
grown = True
nrow, ncol = L.shape[0], L.shape[1]
ntops = len(treetops)
numPixels = Ones(ntops)
Crowns = Zeros((nrow, ncols))
sumHeight = Zeros(ntops)
for i in range(treetops) do
    Crowns[treetops[i][0], treetops[i][1]] = i + 1
    sum_height[i] = L[treetops[i][0], treetops[i][1]]
Crowntemp = Crowns.copy()
while grown = True do
    grown = False
    for i in range(nrows) do
        for j in range(ncols) do
            if Crowns[row, col] ≠ 0 then
                treeid = Crown[i, j] − 1
                seedX = treetops[treeid][0]
                seedY = treetops[treeid][1]
                seedZ = treetops[treeid].height
                mhc = sum_height[treeid]/num_pixels[treeid]
                Neighbors[1] = L[row, col + 1]
                Neighbors[2] = L[row, col − 1]
                Neighbors[3] = L[row + 1, col]
                Neighbors[4] = L[row − 1, col]
                for i in Neighbors do
                    if Extend(i, treeid) == True then
                        Ctemp[i.x, i.y] = Crowns[r, c]
                        numPixel[treeid]+ = 1
                        sumHeight[treeid] = + = i.height
                        grown = True
    Ctemp = Crowns.copy()
return Crowns
```

Table 1: Accuracy for the canopy detection methods

Method	IOU-score	total area	Num canopies
Groundtruth	1	561516.87	12445
canopyDetect	0.8043	475944.99	10994
canopyDetect2	0.2822	207212.67	6789
canopyDetect3	0.9096	537808.37	11451

ground truth. We provided the human annotators with a visualisation of the LiDAR point cloud, the orthophoto and a height map denoting the heights of all the trees in that region. We then asked them to delineate canopies of more than 15 meters in height manually.

The objective of our evaluations was to answer the following questions:

RQ 1 How well does the output of our algorithm match with the human-annotated labels.

RQ 2 Is the total area of tree canopy from the human-annotated labels and the delineation created by the models close to each other?

RQ 3 Is the total number of tree canopies or clusters of trees located close to each other similar in both the methods?

To answer the first question, we calculate the intersection over union(IOU) between the ground truth and the output of the algorithms. For example, if A be the segment obtained as an output to the algorithm and B be the human-annotated ground-truth label, then

$$\mathsf{IOU}(\mathsf{A},\mathsf{B}) = \frac{A \cap B}{A \cup B}$$

The results obtained by the comparison is formulated in table 1.

To answer the second question, we calculate the total canopy area of the delineations given by the algorithms.

We calculate the number of canopies or clusters of trees obtained by the different methods for the third question.

We see that our methods canopyDetect and canopyDetect3 can match with the human-annotated labels with high accuracy. The total area of canopy cover and the number of canopies detected is also relatively close to that of the human-annotated labels. However, our second method canopyDetect2 gets significantly low scores on all of these metrics. The reason may be because the first two methods are based on the Lidar point cloud, which captures the spatial structure of the forest much better than the orthophoto. In the second method, we only use the orthophoto to filter out the trees' heights, while the primary detection is based on the overhead imagery.

We have also implemented canopyDetect3 with alternative tree detction algorithms and the result are tabulated in table 3. We use algorithms by Silva et al and Liu et al for tree detection and evaluate the performance of canopyDetect3 (Silva et al. 2016). We see that Dalponte and Coomes gives the best performance on all the tree metrics.

7 Conclusion

This work proposed three different methods to delineate forest canopy from LiDAR point cloud and orthophoto accurately. While most existing methods rely on instruments on the ground for measuring forest canopy cover in Indonesian peatlands, our model uses LiDAR point cloud data and orthophoto to detect forest canopy cover readily. We showed that our model performs consistently well under extensive evaluations, providing an affirmative answer to the initial question - Can machine learning-based techniques be used for automated detection of forest canopy?

An interesting area to explore in the future is using machine learning-based methods to find and predict vegetation growth patterns in Indonesian peatlands.

Table 2: Accuracy of canopyDetect3 under different algorithms

Method	IOU-score	total area	Num canopies
Groundtruth	1	561516.87	12445
Liu	0.8455	513456.37	10448
Silva	0.8644	526706.37	10945
Dalponte	0.9096	537808.37	11451

References

Dalponte, M., and Coomes, D. A. 2016. Tree-centric mapping of forest carbon density from airborne laser scanning and hyperspectral data. *Methods in ecology and evolution* 7(10):1236–1245.

Dechesne, C.; Mallet, C.; Le Bris, A.; Gouet, V.; and Hervieu, A. 2016. Forest stand segmentation using airborne lidar data and very high resolution multispectral imagery. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 41.

Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X.; et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, 226–231.

Gillies, S. 2013. The shapely user manual. *URL https://pypi.org/project/Shapely*.

Hansen, E. H.; Gobakken, T.; Bollandsås, O. M.; Zahabu, E.; and Næsset, E. 2015. Modeling aboveground biomass in dense tropical submontane rainforest using airborne laser scanner data. *Remote Sensing* 7(1):788–807.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hudak, A. T.; Evans, J. S.; and Stuart Smith, A. M. 2009. Lidar utility for natural resource managers. *Remote Sensing* 1(4):934–951.

Hyyppa, J.; Kelle, O.; Lehikoinen, M.; and Inkinen, M. 2001. A segmentation-based method to retrieve stem volume estimates from 3-d tree height models produced by laser scanners. *IEEE Transactions on geoscience and remote sensing* 39(5):969–975.

Jennings, S.; Brown, N.; and Sheil, D. 1999. Assessing forest canopies and understorey illumination: canopy closure, canopy cover and other measures. *Forestry: An International Journal of Forest Research* 72(1):59–74.

Jim, C. Y. 1989. Tree canopy cover, land use and planning implications in urban hong kong. *Geoforum* 20(1):57–68.

Korhonen, L.; Korhonen, K. T.; Rautiainen, M.; and Stenberg, P. 2006. Estimation of forest canopy cover: a comparison of field measurement techniques.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2980–2988.

Lowman, M. D., and Rinker, H. B. 2004. *Forest canopies*. Elsevier.

Maltamo, M.; Eerikäinen, K.; Pitkänen, J.; Hyyppä, J.; and Vehmas, M. 2004. Estimation of timber volume and stem density based on scanning laser altimetry and expected tree size distribution functions. *Remote sensing of environment* 90(3):319–330.

Neuville, R.; Bates, J. S.; and Jonard, F. 2021. Estimating forest structure from uav-mounted lidar point cloud using machine learning. *Remote sensing* 13(3):352.

Page, S. E.; Rieley, J. O.; and Banks, C. J. 2011. Global and regional importance of the tropical peatland carbon pool. *Global change biology* 17(2):798–818.

Popescu, S. C., and Wynne, R. H. 2004. Seeing the trees in the forest. *Photogrammetric Engineering & Remote Sensing* 70(5):589–604.

Roussel, J.-R.; Auty, D.; Coops, N. C.; Tompalski, P.; Goodbody, T. R.; Meador, A. S.; Bourdon, J.-F.; de Boissieu, F.; and Achim, A. 2020. lidr: An r package for analysis of airborne laser scanning (als) data. *Remote Sensing of Environment* 251:112061.

Rowntree, R. A. 1984. Forest canopy cover and land use in four eastern united states cities. *Urban Ecology* 8(1-2):55–67.

Sačkov, I.; Kulla, L.; and Bucha, T. 2019. A comparison of two tree detection methods for estimation of forest stand and ecological variables from airborne lidar data in central european forests. *Remote Sensing* 11(12):1431.

Saleh, M. B.; Prasetyo, L. B.; Setiawan, Y.; et al. 2020. Evaluation of tree detection and segmentation algorithms in peat swamp forest based on lidar point clouds data. *Jurnal Manajemen Hutan Tropika* 26(2):123–132.

Saremi, H.; Kumar, L.; Stone, C.; Melville, G.; and Turner, R. 2014. Sub-compartment variation in tree height, stem diameter and stocking in a pinus radiata d. don plantation examined using airborne lidar data. *Remote Sensing* 6(8):7592–7609.

Silva, C. A.; Hudak, A. T.; Vierling, L. A.; Loudermilk, E. L.; O'Brien, J. J.; Hiers, J. K.; Jack, S. B.; Gonzalez-Benecke, C.; Lee, H.; Falkowski, M. J.; et al. 2016. Imputation of individual longleaf pine (pinus palustris mill.) tree attributes from field and lidar data. *Canadian journal of remote sensing* 42(5):554–573.

Weinstein, B. G.; Marconi, S.; Bohlman, S.; Zare, A.; and White, E. 2019. Individual tree-crown detection in rgb imagery using semi-supervised deep learning neural networks. *Remote Sensing* 11(11):1309.

Weinstein, B. G.; Marconi, S.; Aubry-Kientz, M.; Vincent, G.; Senyondo, H.; and White, E. P. 2020. Deepforest: A python package for rgb deep learning tree crown delineation. *Methods in Ecology and Evolution* 11(12):1743–1751.

Yin, C.; He, B.; Yebra, M.; Quan, X.; Edwards, A. C.; Liu, X.; and Liao, Z. 2020. Improving burn severity retrieval by integrating tree canopy cover into radiative transfer model simulation. *Remote Sensing of Environment* 236:111454.