Urban Geome-Trees: Automated ~~Modeling~~ reconstruction of Tree ~~Species and Geometry~~ for CFD in Urban Environments

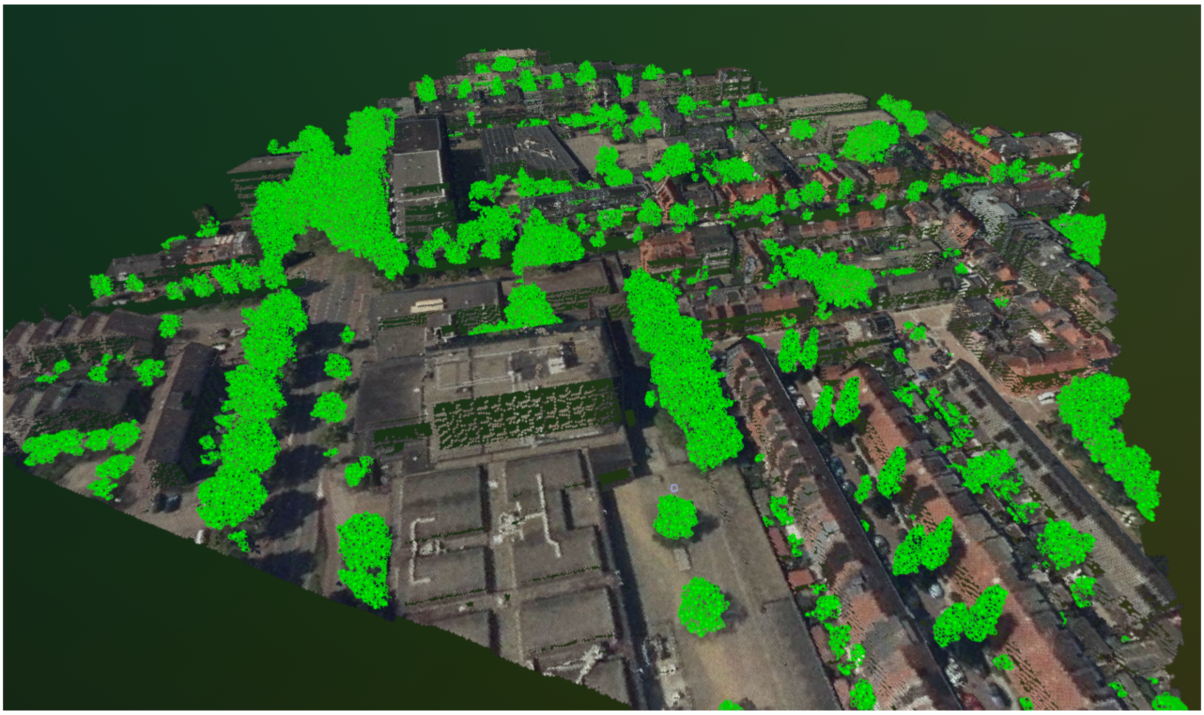
still relevant ?

HL 2025/03/17

Noah Petri Alting

March 17, 2025

student number	4828968
1st supervisor	Hugo Ledoux
2nd supervisor	Clara García-Sánchez



Contents

1 Introduction

1.1 Urban analysis in general

2 -> focus on 3D

- [global overview]

I would start with motivation:

advanced 3d analysis
- wind
- UHI
- etc

are done

using CFD, but often lack trees

1.2 CFD background

- [why CFD]

1.3 vegetation in CFD

- [zoomed in on trees in CFD]

1.4 Get tree species from lidar

?

- [it will help the reconstruction part if not doing alpha wrap but more specific (maybe not needed for CFD]

- [assign porosities based on species]

(havent found a source yet)
(cant it just be data-driven? and skip the species entirely :O)

① add that focus is NL where we have
- lidar
- RGB 8cm
- NIR 25cm

CFD

2 Related work

2.1 How are urban trees currently modelled

Hermann (2024) describes a method to automate extracting tree datasets using LAI, NDVI and mean crown height.
Geert Jan de Groot (2020) describes a method to reconstruct trees from airborne lidar into multiple LoDs. Furthermore, he uses public tree species data from the municipality of Rotterdam.
Lee et al. (2017) describes a method to segment tree instances from lidar. It does not seem to perform well for small trees ($< 20m$) and therefore I will not use this algorithm.

→ OR semantic segmentation, no?

2.1.1 Vegetation indices for tree classification

Chen et al. (2023) describes species classification using deep learning on high resolution UAV RGB and multispectral satellite images. The robustness of the model is questioned still, but the features used to train the model look interesting enough to apply in my research. Table three provides these metrics and the metrics. The best performing features as indicated in the section 3.2 of Chen et al. (2023) are **Norm_G**, **ARVI** and **MTVI2**. I provide their definition in table 1.

Metric	Equation	Reference
NDVI	$\dfrac{\rho_{\text{nir}} - \rho_{\text{red}}}{\rho_{\text{nir}} + \rho_{\text{red}}}$	Haboudane et al. (2004)
Norm G	$\dfrac{\rho_{\text{green}}}{\rho_{\text{green}} + \rho_{\text{red}} + \rho_{\text{blue}}}$	Fraser et al. (2017)
ARVI	$\dfrac{\rho_{\text{nir}} - [\rho_{\text{red}} - \gamma(\rho_{\text{blue}} - \rho_{\text{red}})]}{\rho_{\text{nir}} + [\rho_{\text{red}} - \gamma(\rho_{\text{blue}} - \rho_{\text{red}})]} \quad (\gamma = 0.5)$	Kaufman and Tanré (1996)
MTVI2	$\dfrac{1.5(1.2(\rho_{\text{nir}} - \rho_{\text{green}}) - 2.5(\rho_{\text{red}} - \rho_{\text{green}}))}{\sqrt{(2\rho_{\text{nir}} + 1)^2 - (6\rho_{\text{nir}} - 5\rho_{\text{red}}) - 0.5}}$	Haboudane et al. (2004)

Table 1: Vegetation indices that can be used to identify, separate and classify vegetation.

The Normalized Difference Vegetation Index (NDVI) is a widely used spectral indicator of vegetation health and density based on the difference in red and near-infrared (NIR) reflectance. Healthy green vegetation reflects relatively more NIR light and absorbs more red light. Higher NDVI values therefore indicate active, healthy plant canopies, while lower values suggest sparse or stressed vegetation.

In remote sensing or UAV-based imaging, the normalised green value (norm G) refers to the ratio of the green light reflectance and the total reflected light across the three-band (RGB) image. By normalizing the green band in this way, *normG* adjusts the green band intensity relative to the overall RGB intensity. This approach helps to mitigates the effects of shade, illumination changes and varying exposure, which proofs useful in studies examining burn severity, vegetation health or land cover changes Fraser et al. (2017).

Kaufman and Tanré (1996) introduces the Atmospherically Resistant Vegetation Index (ARVI). The metric provides a vegetation index that corrects for aerosol presence in satellite images. This correction is relevant for satellite imagery but not for aerial imagery. Since I will not be using satellite imagery due to resolution compromises, I will disregard this well performing parameter.

The Modified Triangular Vegetation Index (MTVI2) can be calculated from the reflectance in the near-infrared, red and green band. The MTVI2 value is used for detecting chlorophyll content at the canopy scale while being relatively insensitive to leaf area index Haboudane et al. (2004). For more details check Table 1.

2.2 Tree Instance Segmentation

→ you ought to cite others

Wang et al. (2018) provides a scalable algorithm for tree instance segmentation. The method consists of 5 steps:

1. A preprocessing step that classifies tree points from a raw point cloud.
2. Resampling and clustering, which resamples the imported tree points to cuboid cells and clusters incident cuboid cells in 3D space.

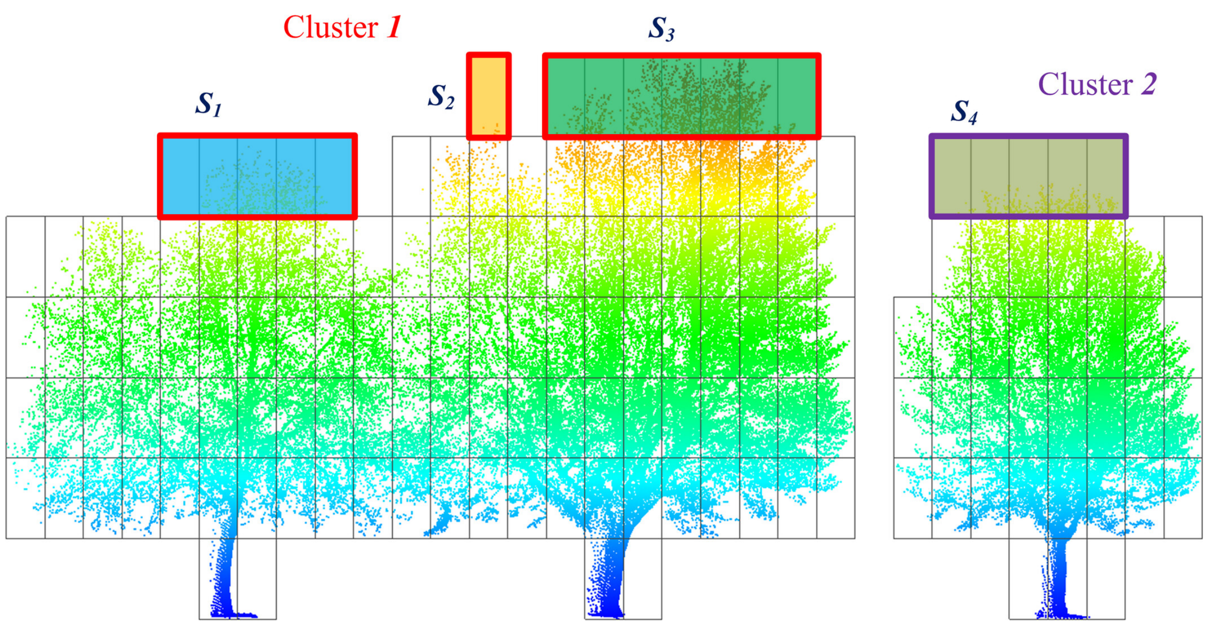


Figure 1: Tree instance segmentation algorithm of Wang et al. (2018).

3. Seed selection over all the clustered cells.
4. Individualise (separate) trees.
5. Evaluation of the overall individual tree delineation quality with respect to manually individualised trees as ground truth data.

Deepforest can be used to detect tree instances based on airborne imagery. It provides prebuilt models for immediate use and fine-tuning by annotating and training custom models on your own data. Its capabilities are discussed in Weinstein et al. (2020).

and that one? what are its pros/cons?

2.3 Leaf Area Index

The leaf Area Index (LAI) is a valuable tool to analyse urban canopy structure. Hermann (2024) has done a significant job in reviewing LAI. Parker (2020) describes many methods and discusses the benefits and drawbacks. Kamoske et al. (2019) was given to me by Ivan and also appears in Parker (2020). It comes with an open source R code to extract LAI, canopylazR.

- [give definition of LAI]

LAI gives an indication about the leave density and it provides insight in where in a tree the leaves are situated. This density can be used for getting the porosity of trees in CFD modelling. Furthermore, since trees have different shapes and leave distribution, I hypothesise that the LAI can be used as a feature to classify a tree species on.

2.4 Inferring species from airborne lidar

this is too vague, why relevant to your project?

- [explain how to go from species to point cloud]

- [use municipality data like Geert Jan de Groot (2020)]

- [LAI-height signature of a tree like Kamoske et al. (2019)]

should be part of a section, not its own section

- [something about the connection to porosity]

2.5 3D geometries for trees in different LoD's

- [meshing trees using geometry + species]
- [alpha wrapping]
- [quality of CFD]
- [assigning porosities to forest model]
- [mention Buccolieri et al. (2018) (urban CFD vs tree review)]

Sect 2

→ I would start with a section about what is necessary for CFD : geometry?
species?
roughness?
etc?

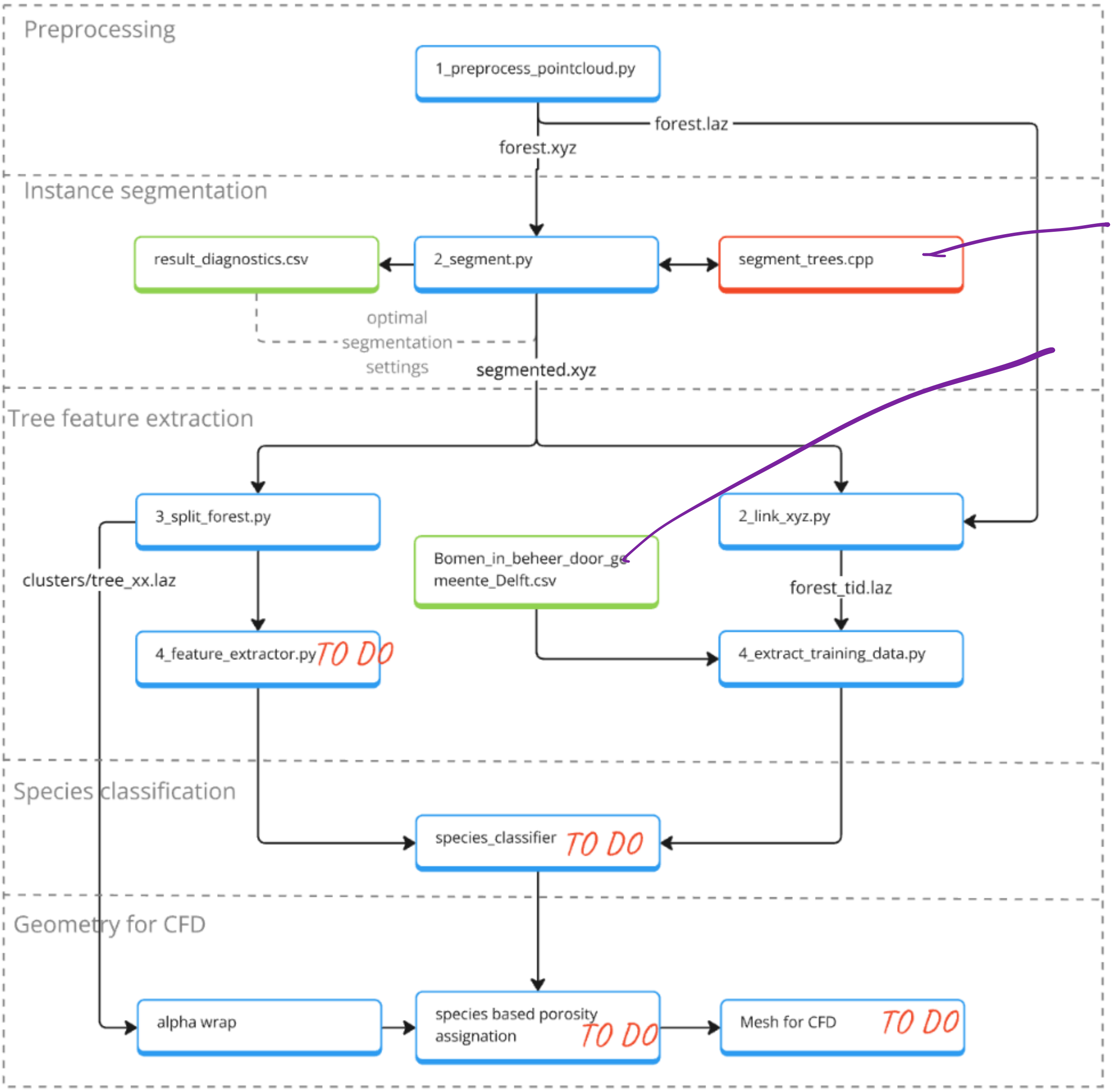
3 Research Goal

not only

This research develops a method to convert point cloud data into CFD-ready tree models, ensuring geometric accuracy and species-specific aerodynamic properties.

ought to be better

4 Methodology



what are the colours?

Figure 2: Pipeline of the full methodology. I propose

4.1 Preprocessing

The point cloud taken from AHN4 contains all points and should be filtered. First the tile taken from geotiles is cut to a region of interest. This is important because a tile from geotiles is too large to process. In my proof of concept case I take a 100m square of the Wilhelmina what if the user wants a large area? impossible?

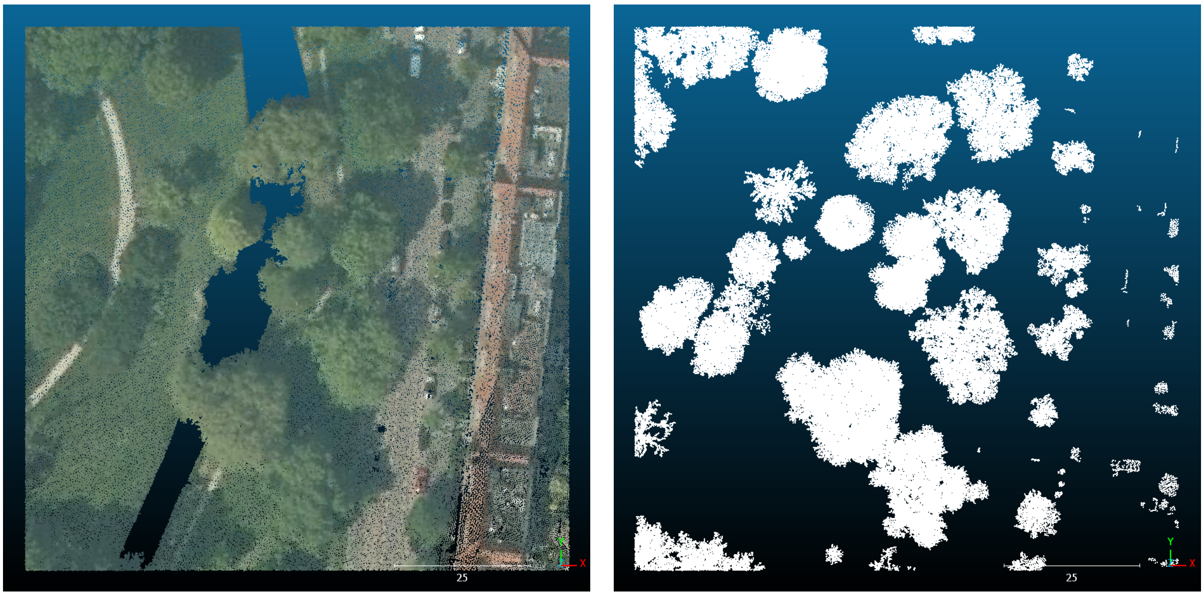
Park in Delft since it is situated near my home and therefore I have some intuition to if a result is close to reality.
The resulting tile contains all points within this geographical bounding box. The biggest and simplest filter to apply is filtering on the number of returns. Since we are interested in vegetation points, we want to filter out buildings and the ground. The AHN point cloud contains the attributes *return number* and *last return*. Each point that is the last return is considered ground or solid building material and is filtered out.

no filter with classification !OC

Subsequently, there might still be outliers in the point cloud. Therefore I also run the statistical outlier removal algorithm provided by the *Open 3D python library*. The parameters for this algorithm are *nb_neighbors* and *std_ratio*. For now, I took the standard values of 20 and 2.0, respectively. Furthermore, it might be good to downsample the point cloud in order to reduce computational load. The resolution of the ANN4 point cloud is 5cm. For tree level analysis, this might be exceed the accuracy needed. However, at this stage the testing area is rather small so downscaling the point cloud would become relevant when scaling up the research to a larger area.

You said it?

After the first filtering steps above, the NDVI value is calculated. Any points that are below a NDVI threshold are considered non-vegetation or dead/unhealthy vegetation. In the testing phase this NDVI value has been lower than expected and seem differing based on sunlit vs shade differences. Due to the unexpected behaviour of the NDVI value, I have not decided on NDVI threshold yet and plan to look further into why this happens before I apply a NDVI filter.
Additionally, the normalised green and the MTVI2 value seem to classify trees well according to Chen et al. (2023) and Haboudane et al. (2004). These values are also calculated in this step of the process, but not yet used as a point cloud filter until I research potential thresholds.



(a) Top-view of the test area (b) Filtered on return number

Figure 3: Top-view of the test area, before and after filtering on return number. It filters non-vegetation by excluding points that are the last return of a lidar pulse and pulses that have only 1 return. The point size in these figures is increased for visibility purposes.

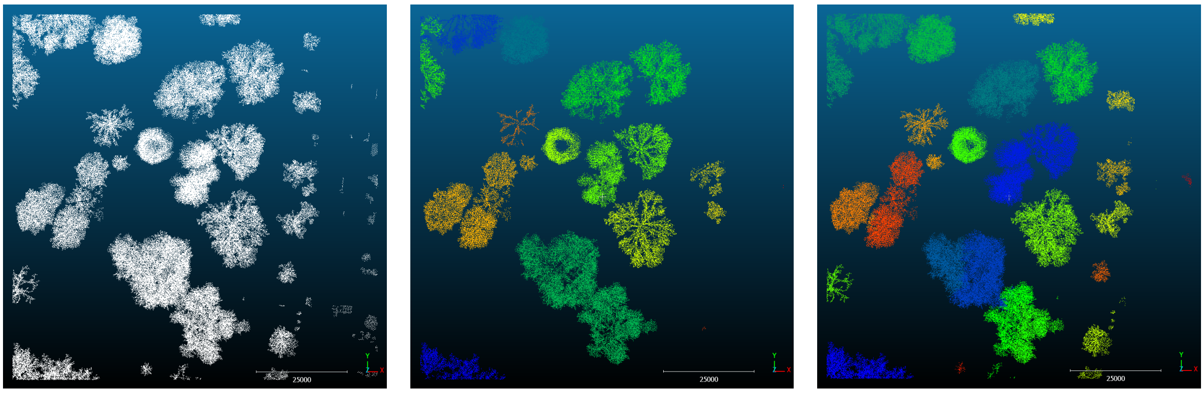
4.2 Tree Instance Segmentation

Wang et al. (2018) provides an open-source, scalable algorithm to segment tree instances. It

— fig would help!

voxelises the point cloud and clusters within voxels per horizontal layer. There are three parameters that dictate the individualization results; searching radius, vertical resolution and minimal points per cluster. — other areas in NL will work?

To tune the algorithm so it works best for the selected area, I ran the segmentation algorithm for different parameter combinations. Afterwards the resulting files containing the segmentation are compared on total number of points and total number of tree clusters. Based on that a set of parameters can be selected. E.g. the amount of points should be as large as possible, ensuring the trees are as complete as they initially were. Additionally, the number of trees found is an indicator of performance. It should match, or be close to the number of trees in the test dataset. Since the test set is close to my home, I was able to visit the Wilhelmina Park to count the trees to see how many the segmentation should roughly output. In this case around 25, with some cases where the truk split very low to the ground. The latter gave me reason to allow those 'twin-trees' to be passed as two individual trees, even though the originate from the same stem.



(a) Top view of the point cloud before segmentation

(b) Segmentation result 0031, having 14 trees

(c) Segmentation result 0028, having 25 trees

Figure 4: Example of the segmentation results. In figure 4b a top view of segmentation case 0031 is provided, having the parameters $radius = 5000$ and $v\_res = 50$ that yielded 14 tree instances. Figure 4c shows case number 0028, which segmented 25 trees using $radius = 5000$ and $v\_res = 200$.

The 100m square area at the edge of the Wilhelmina Park is expected to have over 20 separate tree instances. The exact 'ground-truth' number is difficult to establish since I will allow tree clusters of similar trees to be a single tree after segmentation. At this moment the algorithm seems to work but I will need to tune the parameters using the computing power of the gilfoyle server.

try to separate theory & implementation

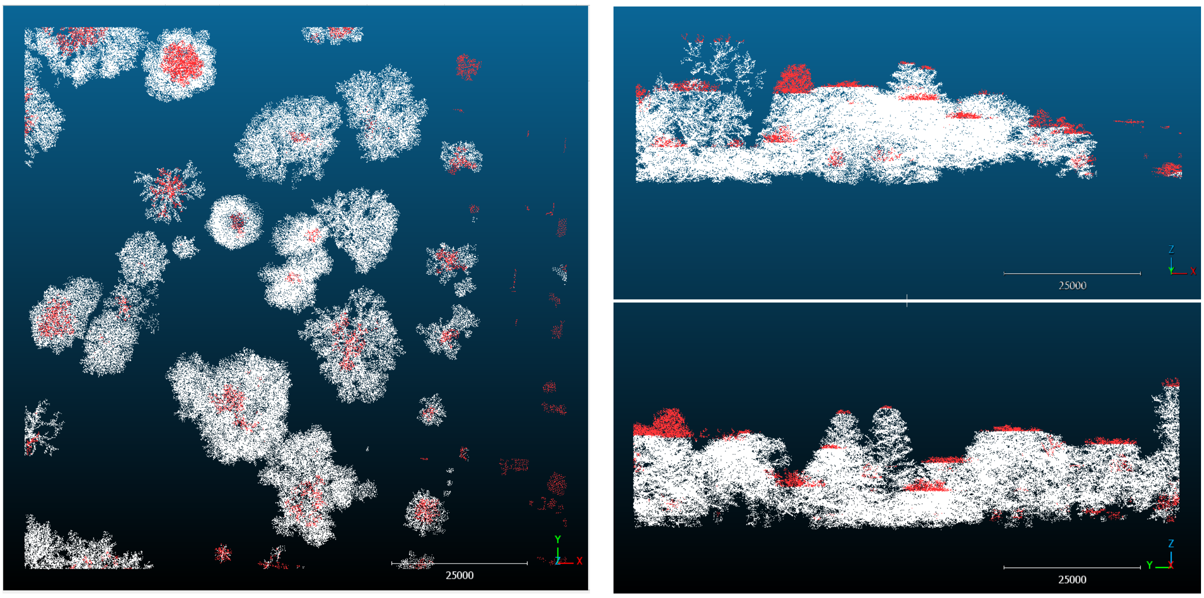
4.3 Tree feature extraction

The instance segmentation code uses XYZ files as input and output. That means that only the coordinates of points are passed after segmentation. Since the coordinates of the points remain unaltered the found tree-id (tid) is reconnected to the point cloud after preprocessing by adding it as a new attribute in LAS format.

The municipality of Delft provided a csv-file containing information about the public trees the municipality maintains. For each tree the file contains, among other attributes, a 2D coordinate in $EPSG : 28992 AmersfoortNew$ and a 'boomsortiment'; species.

To connect the dataset, I take for each tree instance the 2D projection of its convex hull. Then using the geopandas library, a simple spatial join appends a list of all species within the 2D

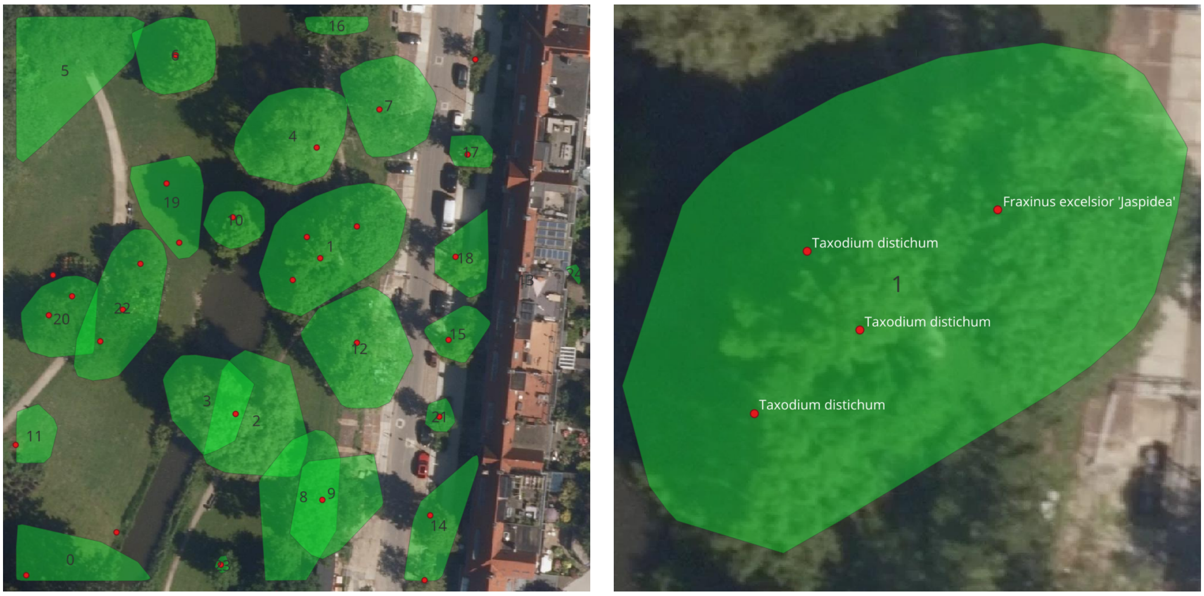
ideally someone should be able to implement your methodology in Rust, or C, or whatever.



(a) Top-view of the unsegmented points. (b) Front and Left view of the unsegmented points.

Figure 5

hull to each tree instance.



(a) Top-view of the test area (b) Zoomed top-view of the tree 1

Figure 6: Convex hulls of the trees after instance segmentation, including the points provided by the municipality of Delft. The tree hulls are shown in green, along with their *tid*. The public trees are red points. In the zoomed view the species is added as a label.

The intermediate step that is given in figure 6 can be used to assess the segmentation step. There are four cases distinguishable for 2D projected convex hull, H_2 and public trees points, t_i:

1. **There is no tree inside the convex hull**, $|t_i \cap H_2| = 0$
   The tree cannot be validated using the municipality maintenance dataset and therefore

no radius given?!

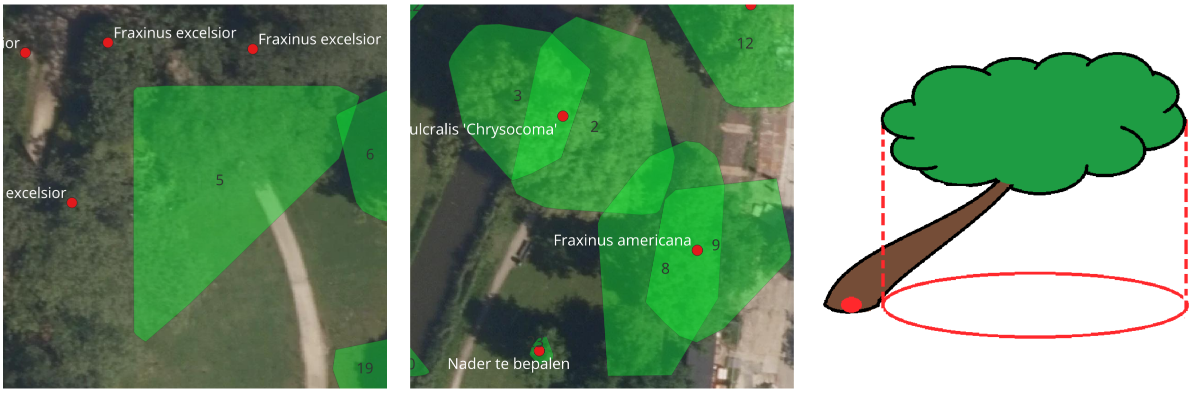
the tree is excluded as training data. A private tree in someone's backyard would be a good example of this case.

2. **There is exactly one tree inside the convex hull,** $|t_i \cap H_2| = 1$
   The tree species found can be assigned to the tree instance as a ground truth species. The species classifier will train that the point cloud belonging to the hull is of said species

3. **There are more than one tree inside the convex hull** $|t_i \cap H_2| > 1$
   The tree instance segmentation algorithm clustered some trees together as one tree. For CFD modelling purposes this is fine as long as the trees are the same species so that the cluster can be assigned an accurate porosity. Therefore the tree cluster is only included in the training data if the species of all trees in the convex hull are homogenous. I still have questions about how to handle this case, since I foresee that the classifier could take the shape of tree clusters into account, e.g. low and wide bush is always species x.

4. **A public tree is shared by** n **convex hulls** $|t_i \cap H_2| = \frac{1}{n}$
   The segmentation algorithm split a tree with one base into two twin crowns. This is fine for the CFD, since both crowns originated from one trunk base, and are then in fact assigned the same species. Ideally, since it is one tree, it is best to merge a twin case back together as one tree. At this point, no triplet (or $n > 2$) is detected, but theoretically this is possible.

There are some cases worth mentioning.



(a) boundary cut (b) twin trees (c) schuine boom

Figure 7: Caption

- [this can be overcome by ...]

4.4 Species classification

- [use classifier to train on tree point clouds]

- [shape and geometrical attributes (from Chi et al. (2025)]

- [individual point features]

- [tree wide features (per layer/percentile/distance to centre]

4.5 3D geometry for CFD simulation

- [create geometry with alpha wrap]

- [potentially taking tree shape into account Geert Jan de Groot (2020)]

- [maybe not important for CFD though, ask clara]

- [porosity based on species]
(no source yet)

- [meshing step between forest obj and snappyhexmesh? ask clara]

5 Time planning

- [make a gantt chart..]

6 Tools and datasets used

- [what to put here? python? qgis? not really very interesting imo]

References

Buccolieri, R., Santiago, J.-L., Rivas, E., and Sanchez, B. (2018). Review on urban tree modelling in CFD simulations: Aerodynamic, deposition and thermal effects. *Urban Forestry & Urban Greening*, 31:212–220.

Chen, X., Shen, X., and Cao, L. (2023). Tree Species Classification in Subtropical Natural Forests Using High-Resolution UAV RGB and SuperView-1 Multispectral Imageries Based on Deep Learning Network Approaches: A Case Study within the Baima Snow Mountain National Nature Reserve, China. *Remote Sensing*, 15(10):2697.

Chi, D., Yan, J., Yu, K., Morsdorf, F., and Somers, B. (2025). Planting contexts affect urban tree species classification using airborne hyperspectral and LiDAR imagery. *Landscape and Urban Planning*, 257:105316.

Fraser, R. H., Van der Sluijs, J., and Hall, R. J. (2017). Calibrating Satellite-Based Indices of Burn Severity from UAV-Derived Metrics of a Burned Boreal Forest in NWT, Canada. *Remote Sensing*, 9(3):279. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.

Geert Jan de Groot (2020). *Automatic construction of 3D tree models in multiple levels of detail from airborne LiDAR data*. PhD thesis, TU Delft.

Haboudane, D., Miller, J. R., Pattey, E., Zarco-Tejada, P. J., and Strachan, I. B. (2004). Hyperspectral vegetation indices and novel algorithms for predicting green LAI of crop canopies: Modeling and validation in the context of precision agriculture. *Remote Sensing of Environment*, 90(3):337–352.

Hermann, T. (2024). *Leaf it to AI: Mapping Urban Tree Morphology and Leaf Area Index with Multimodal Deep-Learning*. PhD thesis, Ecole Polytechnique Federale de Lausanne.

Kamoske, A. G., Dahlin, K. M., Stark, S. C., and Serbin, S. P. (2019). Leaf area density from airborne LiDAR: Comparing sensors and resolutions in a temperate broadleaf forest ecosystem. *Forest Ecology and Management*, 433:364–375.

Kaufman, Y. J. and Tanré, D. (1996). Strategy for direct and indirect methods for correcting the aerosol effect on remote sensing: From AVHRR to EOS-MODIS. *Remote Sensing of Environment*, 55(1):65–79.

Lee, J., Coomes, D., Schonlieb, C.-B., Cai, X., Lellmann, J., Dalponte, M., Malhi, Y., Butt, N., and Morecroft, M. (2017). A graph cut approach to 3D tree delineation, using integrated airborne LiDAR and hyperspectral imagery. arXiv:1701.06715 [cs].

Parker, G. G. (2020). Tamm review: Leaf Area Index (LAI) is both a determinant and a consequence of important processes in vegetation canopies. *Forest Ecology and Management*, 477:118496.

Wang, J., Lindenbergh, R., and Menenti, M. (2018). Scalable individual tree delineation in 3D point clouds. *The Photogrammetric Record*, 33(163):315–340.

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.