3D Forest

User Guide

Release 0.51

A tool for processing of point clouds acquired by terrestrial laser scanning in forests

The Silva Tarouca Research Institute, Pub. Res. Inst.

Department of Forest Ecology.

Authors: Michal Petrov, Martin Krůček, Jan Trochta, and Kamil Král

©2020

Table of contents

Home

Welcome to the 3D Forest wiki/user guide. All information about the installation, usage, and methods implemented within the 3D Forest application can be found here. For more info see the sidebar/table of contents.

1. About Application

The 3D Forest is an application created to process terrestrial laser scanning (TLS) data, point clouds, and to gain detailed information about forest stands and individual trees.

The application is released under the terms of the GNU General Public License v3 as published by the Free Software Foundation. For more info about the license, one can view the LICENSE.txt file located in the program folder or on the web page: http://www.gnu.org/licenses/.

The application is written in C++ and depends on the libraries: VTK, PCL, Eigen, Boost, Flann, LibLAS, and Qt. For a successful installation, users have to build these libraries before compiling the 3D Forest. The source code is provided for downloading and compiling the application on any computer. The Windows installer, as well as trial data, are available on the site: www.3dforest.eu.

2. Detectable Attributes

In the current version of the 3D Forest (0.51) the following characteristics can be calculated:

Tree Attributes

- Position: gives X, Y, Z coordinates of the tree base in the Cartesian coordinates system. More information in section Trees.
- DBH: this attribute determines diameter at breast height, i.e. the tree stem diameter calculated from a sub-set of points from 1.25 to1.35 m above the tree base. See also: least squares regression and randomized Hough transformation.
- Height: vertical distance between the tree base and the highest point of the tree, (i.e. the max. difference in Z coordinate in meters).
- Cloud length: gives the longest distance between two points in the cloud. It is suitable for the length calculation of highly inclined or lying trees.
- Stem Curve: determines the stem centers and diameters calculated in various heights above the tree base (0.65m, 1.3m, 2m, 3m, etc.).
- Convex planar projection of the tree: returns polygon with the shortest boundary containing all points of the tree cloud orthogonally projected on the horizontal plane.
- Concave planar projection of the tree: returns polygon with the smallest area containing all points of tree cloud orthogonally projected on the horizontal plane.
- The number of tree points: yields the number of points representing a single tree.

Crown Attributes

- Crown centroid: gives coordinates of the tree crown center position (X, Y, Z) in the Cartesian coordinates system as computed from crown external points. More information in section Crowns.
- Crown position deviation: returns the crown position deviation from the tree base position. It is defined by distance (m) and direction (°).
- Crown bottom height: provides the vertical distance (i.e. the difference in Z coordinates in meters) between the tree base position and height of the place where the lowest living branch attaches the main stem.
- Crown height: computes the vertical distance (i.e. the difference in Z coordinates in meters) between the crown bottom height and its highest point.
- Crown total height: gives vertical distance (i.e. the difference in Z coordinates in meters) between the lowest and the highest point of the tree crown.
- Crown volume by voxels: returns crown volume computed from voxels of a given size.
- Crown volume and surface area by concave polyhedron: computes crown volume and surface area utilizing cross-sections of a given height and concave hull threshold distance.
- Crown volume and surface area by 3D convex hull: returns volume and surface area of crowns 3D convex hull.
- Crown intersections: gives volume and surface area of intersecting space between convex hulls of two crowns.

3. Installation

WIN

The 3D Forest installer for Windows is located on https://github.com/VUKOZ-OEL/3DForest/releases/download/v0.5/3DForest_05.exe. Once downloaded, the installation wizard will guide you through. Furthermore, sample data to test the 3D Forest are placed on https://github.com/VUKOZ-OEL/3dforest-data. For uninstalling run the uninstall.exe file located in the installation folder.

WIN building from source code

There is an option to download and compile the source code for Windows too. In such a case, additional libraries and compiler are needed as prerequisites: VKT, PCL, Qt, Eigen, Flann, libLAS, and for example MinGW64 cross-compiler. At the moment, there is a modified version of tpoint cloud library (PCL) for point and area picking in the selection mode.

For installation from the source, it is important to install dependencies first. It is also important to have compilator, IDE and GIT installed.

Let's start with Visual Studio, and VCPKG package manager.

- first download vcpkg: `git clone https://github.com/microsoft/vcpkg.git`
- build vpckg again: `.\bootstrap-vcpkg.bat`
- find the right version of dependency - problem is with VTK and Qt those will be installed separatelly. -Finally install first round of dependecies: `vcpkg.exe install eigen3 boost flann liblas qt5 --triplet x64-windows`
- `vcpkg.exe integrity install`

FLANN

Even you installed Flann library in previous step, you will need to use different version in vcpkg directory try to find all versions `vcpkg x-history flann` and try to use version 1.9.1 `git checkout c626675abb963d15f5d290a56005556d95b160bd -- ports/flann` after this remove flann `vcpkg remove flann` and install back `vcpkg install flann`

VTK

Since vcpkg has some isssues with vtk versions, vtk should be compiled separatelly from source.

- get source `https://www.vtk.org/files/release/8.2/VTK-8.2.0.zip` using CMAKE configure dont forget use with Qt and install using i.e. VS project. I had problem with QpaintPath, so just include in into file.

Install PCL From source

- `git clone https://github.com/janekT/pcl.git`
- open in Visual studio configure,build and install

Install all from source

- QT: `git clone https://code.qt.io/qt/qt5.git qt5`
- BOOST `https://dl.bintray.com/boostorg/release/1.73.0/source/boost_1_73_0.zip`
- eigen: `git clone https://gitlab.com/libeigen/eigen.git`
- flann: `git clone https://github.com/mariusmuja/flann.git`
- VTK: `https://www.vtk.org/files/release/8.2/VTK-8.2.0.zip`
- libLAS: `git clone https://github.com/libLAS/libLAS.git`
- PCL: `git clone https://github.com/janekT/pcl.git`
- 3D FOREST: `git clone https://github.com/janekT/3DForest.git`

macOS, Linux

For the macOS and Linux users, there is only an option to download and compile the source code.

At first, dependencies are installed using `brew install boost vtk qt flann Eigen LibLAS Cmake`. The second step is to compile PCL from the source as 3D Forest uses unapproved changes in code. It can be downloaded from https://github.com/janekT/pcl/tree/pointpicking using git or as a zip file. Cmake creates a project for Xcode. The PCL Library has to be compiled and installed then.

The 3D forest can be downloaded from the repository https://github.com/VUKOZ-OEL/3DForest/tree/master. CMake finds all dependencies for the configuration of the project. The 3D forest can be run via Xcode.

Linux installation is done by the package manager similarly to MacOS.

4. Brief Insight and Control of the Application

The control toolbars and menus are on the top of the application window. Toolbars are divided into logical sections – project bar with opening, creating new, importing of projects, etc. The next toolbar affects the view of the clouds in the visualization area. Six icons represent six predefined views of displayed clouds. These are followed by two icons representing the change of the view – orthogonal and perspective projection. The tree and crown toolbar serves for displaying computed attributes. There is a list of all clouds imported into the project on the left side of the application. The biggest part of the application win\sdow is the visualization area where selected clouds and tree/crow variables are visualized. There is a status bar where information about selected methods or clouds is displayed. In the right bottom corner is displayed progress bar when some action takes place.

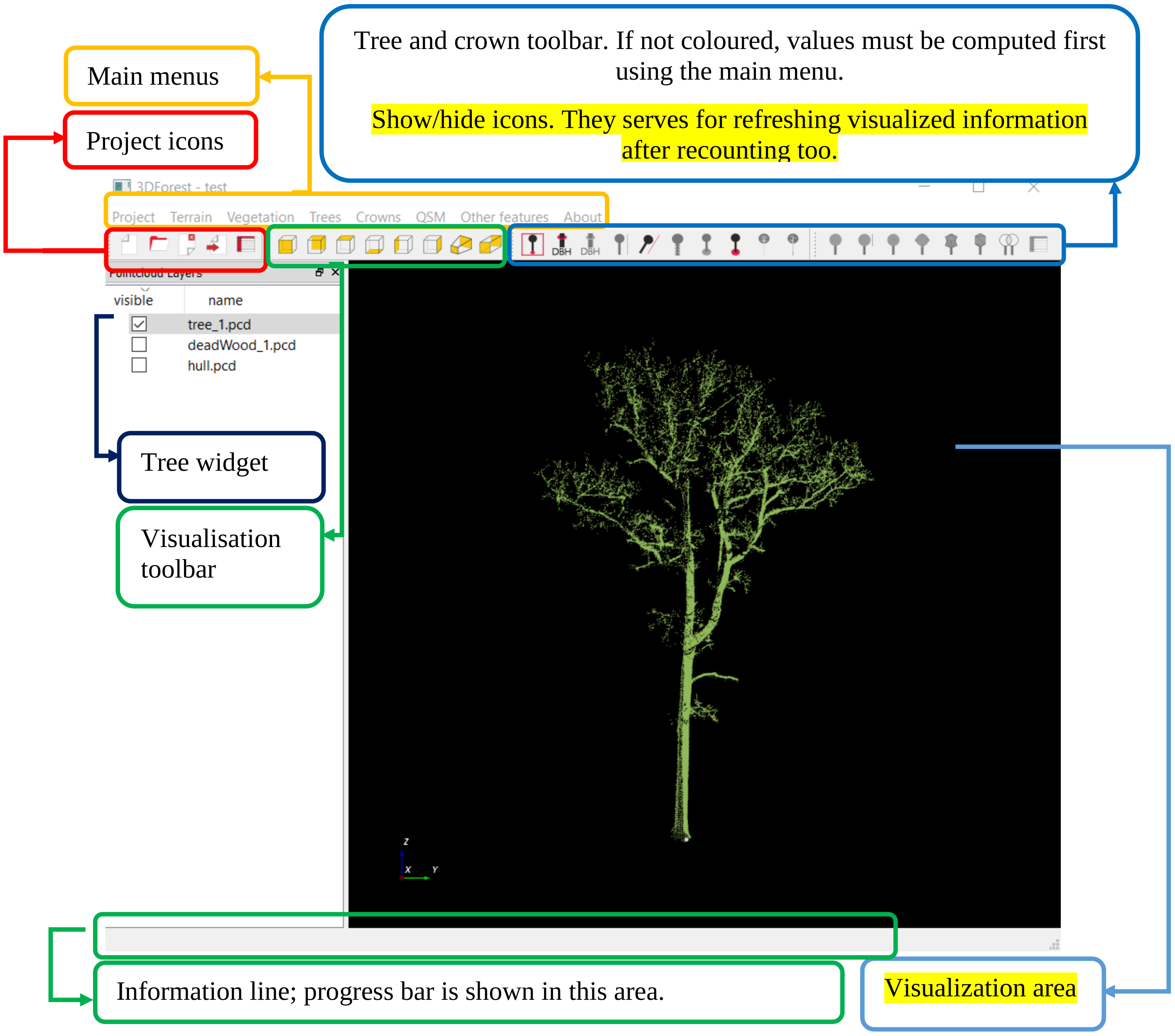

Main menus
Project icons
Tree and crown toolbar. If not coloured, values must be computed first using the main menu.
Show/hide icons. They serves for refreshing visualized information after recounting too.
3DForest - test
Project Terrain Vegetation Trees Crowns QSM Other features About
Pointcloud Layers
visible name
tree_1.pcd
deadWood_1.pcd
hull.pcd
DBH
DBH
Tree widget
Visualisation toolbar
z
x y
Information line; progress bar is shown in this area.
Visualization area

Figure 1: The application window and description of its parts.

Basic control

Mouse button configuration in the visualization area:

Left-click: draw to rotate the view

Middle-click: draw to zoom the view

Middle rotate: zoom the view

Right-click: draw to move the view

Shift + left click: information about trees at the selected point

Ctrl + left click: clockwise or counter-clockwise rotation

Context menu on cloud list (right-click on cloud name):

Delete: delete cloud from the project and if needed from the disk

Color: change the color of a cloud

Color by field: set the color of points in the cloud by selected field (axis x, y, z, and intensity)

Point size: change the size of points in the cloud

All Clouds ON: all clouds in the project are visible

All clouds OFF: all clouds in the project are invisible

Keyboard

p, P: switch to a point-based representation

w, W: switch to a wireframe-based representation (where available)

s, S: switch to a surface-based representation (where available)

+/ -: increase/decrease overall point size

g, G: display scale grid (on/off)

u, U: display lookup table (on/off)

r, R: reset the camera

f, F: focus on the point

x, X: selection (in selection mode)

o, O: perspective (on/off) Alt +: show and allow quick access keys in the menu (underlined character)

Tips: Switching between wireframe and point-based representation does not affect the point clouds. The correct values of the scale grid are shown only when the perspective is off. The display lookup table shows values extent and color range if Color by field is used.

Cloud type description

All data imported into the 3D Forest are stored in the PCD file format (binary compressed PointCloudData). PCD file contains coordinates of all points with intensity value. These four variables of the point representation are being used for the moment. Additional information such as color or path to the transformation matrix file can be found in the MyProject.3df file.

There are different point cloud representations during 3D Forest data processing:

Base cloud:

Represents raw data imported into the 3D Forest. The data should be preprocessed in some other software (usually provided with the scanner), where cloud fitting and registering are made. The base cloud is not differentiated. It embodies points of all objects such as terrain, vegetation, buildings, etc.

Figure 2: Example of the base cloud before any segmentation – all points contain just X, Y, Z, and intensity values; no objects or surface types are differentiated.

Terrain cloud:

Using Terrain analysis the base cloud can be divided into vegetation and terrain – two main parts of the forest ecosystem. The terrain cloud represents ground surface and can be used for better calculation of tree position or exported into GIS software for detailed (micro)topography analysis.

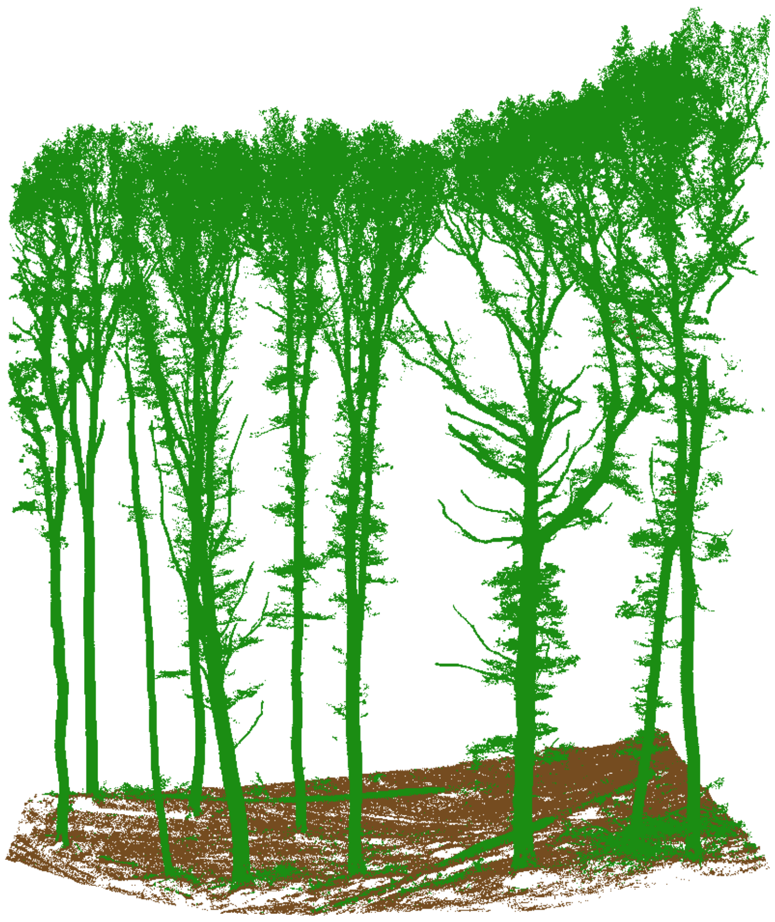


Figure 3: Visualization of Terrain cloud (brown) and Vegetation cloud (green) after ground/vegetation separation.

Vegetation cloud:

The part of the base cloud which is not terrain is labeled as vegetation. From this point cloud, the individual trees can be segmented automatically or manually selected so to create another cloud type – tree cloud.

Tree cloud:

This cloud represents a single tree after manual segmentation. Only for tree clouds, it's possible to compute basic tree variables like DBH, height, position, etc.

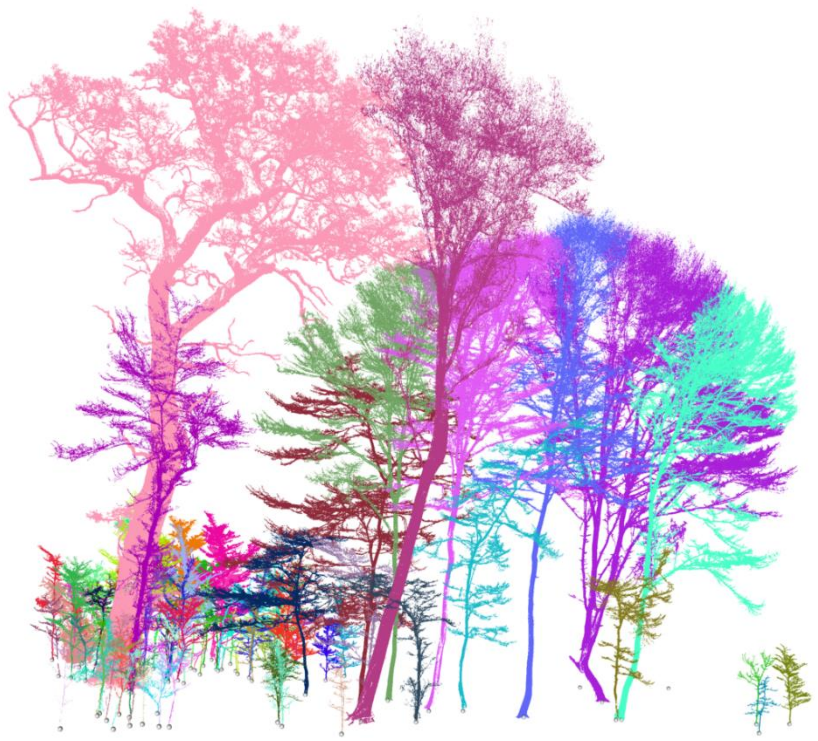


Figure 4: Visualization of Tree clouds after tree segmentation – individual trees are displayed in different colors.

Other:

Clouds that represent unclassifiable points or points which do not belong to any other cloud type. These clouds can be displayed or used as vegetation for tree selection/segmentation.

5. Basic Workflow

Once the application is installed, the workflow is to set up a project, import data, and segment data to terrain, vegetation, and trees. Only then the tree and crown features can be evaluated.

Start a Project (Project Menu)

The first step when using the 3D Forest is to set up a project. This can be done via the Menu -> Project -> New Project. In this step, a directory of the whole project can be selected. This directory then becomes a place for data storage. The transformation matrix, i.e. the data extension, can be set at this point.

Import Data (*Project Menu*)

Once the project is set, the raw data can be imported. It is essential to import them as the base cloud if they consist of all points of the forest (terrain, trees, buildings, etc.). In case the data was already separated into terrain and vegetation, each point cloud has to be imported separately according to its type (terrain or vegetation). Likewise, the clouds of individual trees should be imported as tree clouds.

Visual Check

It is important to check the cloud quality after the import. Mainly if there are any doubts during setting-up a project. If the cloud looks like parallel lines, the transformation matrix is wrong. Then the project has to be recreated with a different (appropriate) transformation matrix.

Separate Ground and Vegetation (*Terrain Menu*)

Once the base cloud is imported, it needs to be divided into two basic parts: terrain and vegetation. Two alternative methods are provided to fit this request. Terrain from octree, which returns ground points of cloud. And terrain from voxel, which gives a voxelized cloud represented by centroids of original points in each voxel. All points that do not belong to the terrain cloud can be manually selected and removed during a visual check. You can also use some of the implemented filtering methods.

Analyze Terrain (*Terrain Menu*)

There are functions to clean terrain point cloud using the following filters: statistical outliers filter and radius outlier filter provided by PCL. Besides, there are functions for interpolation of missing terrain using inverse distance weighting (IDW) if gaps are present, and functions for evaluation of slope, aspect, curvature, hillshade, and a function that helps to identify features within the terrain point cloud.

Segment Vegetation (*Vegetation Menu*)

For a tree analysis, it's essential to work with clouds that represent single trees. This is achievable by automatic or manual segmentation of the vegetation. During manual segmentation, all the points that do not belong to the tree have to be removed before saving a single tree cloud. On the contrary, automatic segmentation treats the whole vegetation cloud according to the input parameters. A more detailed description of the automatic segmentation method is presented in section Vegetation.

Edit a Tree Cloud (*Trees Menu*)

If needed, the tree cloud edit helps to manually remove points that do not belong to the particular tree. It also enables to merge a tree separated into two clouds.

Analyze Trees (*Trees Menu*)

The 3D Forest is designed for individual tree analysis. Thus the first attribute to be evaluated is the tree position (defined as the tree base position). This is required for the estimation of all other tree variables like tree DBH, height, length, stem curve, or tree planar projection.

Analyze Crowns (*Crowns Menu*)

TLS is a great tool for scanning objects in remote distances, e. g. a tree crown. However, the analysis of tree crown requires the position and tree height to be calculated first. Unless this is done, the crown analysis menu is inactive in the Crowns Menu (greyed out). Besides, the crown analysis has to be computed using the cloud that consists of points that belong to the tree crown only, i.e. not to the stem. There are both manual and automatic segmentation procedures to do that. Once the crown is segmented from the tree, crown features (crown height, crown length, crown width, crown centroid, crown displacement, crown voxel volume, crown convex 3D hull, crown concave hull, or intersection of crowns) can be evaluated.

Quantitative Structure Model (*QSM Menu*)

The last group of attributes is listed under the tree quantitative structure model (QSM). It gives information about stem and branches (length, connection). As well as for the crown analysis, the position and tree height has to be evaluated first.

6. Setting a Project

New project set up

Setting up a new project is done via the *Menu -> Project → New Project*. Also, there is the New Project icon in the project context menu. In the project manager window, set the name of your project (without spaces) and select the path where to save it. In this location, a new folder with the project name will be created and all files of the project will be stored there, i.e. project base file (MyProject.3df) and all imported clouds. The next step is to set a transformation matrix. There is only one transformation matrix for each project to be employed. The transformation matrix serves for reducing the number of digits in coordinate values for faster data management (RAM and cache-friendly administration). To create a new transformation matrix it's necessary to know the exact extent of your data coordinates. It is also possible to select from pre-prepared matrices or create a new one. For creating a new matrix set name of matrix, and values of transformation. The goal is to set the transformation matrix as a number as big as possible so that subtracting this number from your coordinates do not change the data extent that carries information. There is an example in table 1. There is also the NO_MATRIX option if doubts about data extension. Data are transformed into the new coordinate system automatically during import, only for .pcd files there is a possibility to decide if transformation gets involved. Correspondingly, exported data from 3D Forest are transformed back into the original coordinate values.

Table 1: An example of the data before and after transformation.

	Original data extent	Transformation matrix	Data extent after transformation
max. x	18857748,76	-18857100	648,755
min. x	18857104,96		4,96
max. y	-987657123,6	987656800	-323,55
min. y	-987656816,6		-16,55
max. z	425,33	0	425,33
min. z	358,13		358,13

Project Opening

A project can be opened via *Project→ Open Project*, or click on the Open Project icon in the project context menu and select your project file (file with extension .3df).

Project Import

Since projects save their path, simple copying into another location (disc or directory) would break the project. If such a change is needed, it has to be done as an import of an old project into a new location. Go to the *Project → Import Project*. Choose the path to the old project, set the name of the new project and location where the new project will be created. The new folder with the project name will be created and all data files will be moved there. It is also possible to select the option for removing the old project from disk after the import.

Data Import

The 3D Forest enables to import data into a project via *Project → Import*. It is possible to import data in following formats: .txt, .xyz, .las, .pts, .ptx. If the user's data already have the 3D Forest native format (PCD), it is important to choose an appropriate cloud type (see Cloud type description) and to set the transformation matrix. Once the data are imported, they are listed at the main window and in the list of project clouds.

7. Terrain analysis

There are two automatic methods for terrain processing. None of these methods are perfect, so it is advised to use both of them to combine the advantages of their outputs. There is also a possibility of manual adjustment of the results.

Terrain from octree

Go to *Terrain → Terrain from octree*. In the dialog window select the base cloud that should be processed. Enter the name of the new terrain cloud, the name of the cloud where the rest of the data (usually vegetation) will be stored, and set the resolution (length of cube edge in cm). The input cloud is divided into cubes. Cubes containing points and having the lowest z-value are considered to be the "ground cubes". The terrain is then defined by the points in these ground cubes. This means that all the points in ground cubes are considered to be terrain and contrary to voxelization there is no reduction of points and no generalization. This method is more laborious for manual post-processing. However, the results are more detailed and built by original points.

HINT: If there are points that belong to the terrain within the vegetation cloud, i.e. points connecting trees after the segmentation, it's recommended to use a higher resolution.

Terrain from voxels

Go to *Terrain → Terrain from voxels*. In the dialog window select the base cloud to extract the terrain. Enter the name of the new terrain cloud, the name of the cloud containing the rest of the data, and set the resolution of voxelization (length of voxel edge in cm). Appropriate resolution to gain relevant results is below 50 cm. The input point cloud is reduced into voxels and centroids of voxels (i.e. points with averaged coordinates from all original points of individual voxels). Voxels with the lowest z-values define the terrain. Increasing voxel size also rise the overestimation of terrain height (z-value of centroids is affected by more points which lie above the terrain). However, as the resulting cloud contains fewer points than the cloud produced by Terrain from the octree also manual post-processing is usually faster.

Statistical outlier removal filter

Both terrain cloud from voxels or form octree may be filtered using statistical outlier removal. Go to *Terrain → Statistical outlier removal*. In the dialogue window select terrain cloud set. Name the cloud with removed points, and set the number of neighbors for computing mean distance and standard deviation. Points with a mean distance longer than standard deviation are removed. For more details see the PCL documentation.

Radius outlier removal filter

Applying radius outlier removal filter to terrain cloud is possible via *Terrain → Radius outlier removal*. In the dialogue window select terrain cloud set. Name the cloud with removed points, set the radius, and the number of neighbors. Points with fewer neighbors in a given radius are removed. For more details see the PCL documentation. Tips: If too many or too few points were removed by filters use the cloud merge feature and repeat the procedure.

For the best result combination of both filters is recommended. Note that different voxel sizes and various initial cloud density (if Terrain from octree is used) need different filters settings. Both filtering methods are better to use with cloud created by Terrain from voxels due to its regularity.

Manual adjustment

The above-described terrain extraction can be also adjusted by manual editing. Go to *Terrain → Manual adjustment* and select terrain cloud for editing. Set a name for a new cloud where points removed from the terrain cloud will be saved. Pressing the "x" button activates the selection box. The left mouse button makes the selection. The designated points will be removed from the terrain file and stored in the separate file defined above. For a step back press the undo icon → in the top panel ↶. There is also a split function that creates user-defined strips of terrain for easier point selection. Arrows switches among the strips. The strip selection is finished by clicking at the split function again. Stop EDIT icon ⬇ saves all changes in the original (input) terrain cloud.

IDW interpolation

Areas with missing terrain points (typically shaded by thick trees or located right under the scanner during scanning) may be filled in by Inverse Distance Weighted interpolation (IDW) from the surrounding terrain points. Go to *Terrain → IDW* to select an input terrain cloud for interpolation. Set a resolution of interpolation in cm, a number of the closest points (n) to be included in the interpolation. Name the new interpolated terrain cloud that will be free of empty areas. The interpolation uses n-closest points of the original terrain to estimate Z value of the new terrain point.

Slope

The slope analysis is a method that evaluates the terrain slope. This function is reachable via *Terrain → Slope*. The terrain slope, in general, is defined as the first derivative of the terrain. The slope is thus calculated from the height difference based on a user-defined sector. Users can choose between the sector defined by the nearest neighbors or the radius. Each point of the selected terrain cloud is then compared to the others within the defined sector and the mean slope value is computed. The result is stored as a field of intensity in degrees or in percent, where 45 deg represents 100%.

Aspect

The aspect is a method for evaluation of the slope orientation for a given terrain layer. There is defined surrounding for each point of the terrain layer where the average orientation is calculated. The result is given in degrees, i.e. -180 to -135 = West, -135 to -45 = South -45 to 45 = East 45 to 135 = North 135 to 180 = West. The result is stored as a field of intensity.

Hillshade

This function returns shaded relief of a given layer of terrain. The values of the resulting cloud range from 0-255 according to the azimuth and orientation.

Terrain Features

This is a method that looks up identical places in the field based on their parameters. Identical places are defined by their value in the field of intensity, the number of points, the size of the major axes, the axes ratio, the size of the maximal area (the concave), the convex, and the convex/concave ratio. The result is not saved to disk nor project as there is the Export Features function to do it.

Feature table

The feature table displays a table of attributes for each terrain feature.

Export Features

This function exports the terrain feature information. The user selects the directory to save the file, delimiters of fields, and files to export. There is an option to choose between an attribute file and Convex/concave polygon. The attribute file is a file with the position of the centroid of selected voxels and its attributes. Convex/concave polygons are polygons delimiting voxels of given features. These are saved in the VKT format.

8. Vegetation

After the terrain extraction, there are two types of clouds in the project: the terrain and the vegetation clouds. However, the vegetation cloud needs to be further segmented into single trees to get the desired information about tree attributes. Right now there are two methods implemented in the 3D Forest, automatic and manual segmentation.

Automatic segmentation - version 0.5 and higher

The automated segmentation is called via *Vegetation → Automatic tree segmentation*. There are ten inputs needed to start. The vegetation and terrain clouds of interests are obvious. The voxel size, descriptor type, descriptor threshold value (%), amount of iterations, number of voxels in element, and the distance from terrain are explained in detail in the next paragraphs, as well as the method itself. The cloud prefix and non-segmented points output name define the output.

The algorithm used in version 0.5 and higher is based on searching neighboring voxels according to the chosen descriptor. At first, voxels of a given size (voxel size in cm) are made through the whole vegetation cloud. Each voxel is then evaluated by user-defined descriptor: Principal Component Analysis (PCA), slope, intensity, and PCA-slope multiplication.

PCA computes Principal Component Analysis of x,y,z coordinates of all points inside each voxel. The ratio of different PCA axes is then used as the descriptor. Based on our experiences, the PCA descriptor usually gives the best segmentation results for TLS data.

The threshold value is defined as a percentile of the descriptor value. The range is 0-100% and represents the lower limit of the used voxels for segmentation. For example, if the input threshold value is 70 then only the voxels whose descriptor value is at 70% of the actual value range or higher are used for the tree extraction. The higher the threshold is, the more similar and compact surfaces are represented. The ideal values of the input parameters are higly dependent on the point density and structure of the forest stand (see Tips below this paragraph). Resulting representations are the surfaces of a stem or big branches.

All voxels above the descriptor threshold value and within the voxel size are then grouped. This means that these groups of voxels contain points of similar topological characteristics and the voxels are neighbors (connected without interruption). Such groups of voxels are called elements if they satisfy the condition of a minimal number of voxels in an element. Within these elements, the tree bases are classified by complying with the given distance above terrain points.

As branches or stems may be interrupted by occlusions bigger than the voxel size and/or small sprigs do not always fill the descriptor threshold value, it is needed to add more voxels to make the tree complete. The established tree bases are thus gradually connected with voxels forming the rest of the tree parts during the iterative process. This allows us to find all free voxels that are onward from the voxel size.

The iterative method is automatic and has two parts. It starts by omitting the descriptor condition. Thus all voxels that are up to the voxel size are considered as belonging to the tree. The surrounding of every added voxel is instantly searched for the neighbors. Once all neighbors are added, the voxel size is increased by its value, i.e. plus the voxel size, and the new surrounding of established tree bases are searched again. If the free voxels or elements within this space have the descriptor value up to the threshold they are added to the tree.

Afterward, all free voxels laying within the doubled voxel size are added to the enlarged tree as well as their neighbors. These two steps are repeated iteratively according to the number of iterations. The appropriate number of iterations thus depends on your data quality and forest characteristics. The data with minor occlusions can be segmented well with few iterations. On the other hand, too few iterations can lead to omission errors (missing treetops), too many iterations can lead to commission errors, especially in very dense forests.

When the iteration is finished, segmented trees are saved into the project folder as separate clouds with a given prefix and number of the tree. All points that are not selected as a tree are saved in vegetation as a "rest cloud".

Tips:

- Voxels should have a size to incorporate at least 3 points from the point cloud. If the point cloud is dense enough, a smaller voxel size can be used and vice versa.
- For dense forest stands smaller voxel size is advised to maximize the number of segmented trees. On the other hand, with small voxel size trees might not be complete after segmentation.
- For sparse/even forest stands larger voxels can be used.
- If neighboring trees are merged after segmentation, try to increase the threshold of the descriptor value or/and use smaller voxels.
- If you want to avoid segmentation of small trees increase the distance from terrain and voxel size.
- It is better to use fewer iterations for deciduous trees scanned in the leaf-off state (lower occlusion is anticipated). On the contrary, for coniferous (evergreen) trees usually, more iterations are required due to occlusions in crowns.
- In some cases, two iterations of segmentation can be useful, first run the segmentation with finer parameters. Then segment the *rest-of-vegetation* point cloud with coarser parameters.

Manual segmentation, segmentation of alternative vegetation

Manual tree segmentation or segmentation of additional vegetation is done via *Vegetation → Manual tree selection*. The source cloud with vegetation can be selected in the dialog window as well as the name of the cloud where the rest of unsegmented points will be saved. The name of the cloud with a segmented tree (i.e. the tree cloud) is set after actual segmentation. The segmentation itself is done via the "x" button that activates the selection mode. The selection box is drawn using the left mouse button. Selected points are removed from the view. There is an "undo" key if needed. The goal is to remove all points that do not belong to the target tree. After editing, i.e. the target tree is shown only, "Stop EDIT" icon ends editing mode. The filename of the new tree cloud has to be set. Once the segmented tree is saved, another segmentation process can be started (all the removed but unsegmented points will reappear) or terminated. Some points of vegetation belonging to trees, e.g. isolated branches, may remain in the unsegmented vegetation file after segmentation. These points may be attributed to any of the segmented trees by the "Cloud merge" function.

Tips:

- Larger areas are recommended to split into smaller tiles for improved visibility and easier handling. This may be done by employing additional software, e.g. LAStools.

- The segmented trees are removed from the edited vegetation cloud. This leads to gradual simplification. Thus, the easy shaped and/or stand-alone trees are suggested to be segmented first and the complex parts afterward.
- If the tree position and DBH are the only required variables, it is not necessary to segment the whole trees. For these variables, it's enough to separate the bottom part of the stem only (approximately 2 meters above the terrain).
- The neighboring tree clouds, and/or the vegetation cloud can be displayed by checking the box in the tree widget. This may help to decide on the controversial points.

9. Trees

The *trees* is a part of the 3D forest menu that serves for computing attributes of individual trees or tree cloud editing.

Tree Cloud Edit

To remove points from an existing tree cloud, go to the *Trees → Tree cloud edit.* The dialog window gives options to choose the tree cloud to be edited and to set a name for a new cloud with all removed points. Points are removed by the same scheme as in the manual segmentation, see section Vegetation.

DBH Cloud Edit

If the results of DBH (Diameter at breast height) estimation are suspicious, there is an option to display and edit the calculation source point cloud. The path *Trees → DBH cloud edit* enables the selection of the tree and the respective DBH cloud to be edited. The editing scheme is as in the manual segmentation again. All selected points will be excluded from the DBH computation. However, they will be removed from the tree cloud temporarily, i.e. for the 3D Forest session. The DBH cloud edit is voided by closing the project.

Tree Base Position

Tree base position is a key variable providing a baseline for computation of other tree variables such as DBH, tree height, stem curve, and the visualization of convex/concave tree projection. Therefore, these functions are not available before the tree position is calculated. The 3D forest offers two methods for tree base position calculation. However, their results may slightly vary (Fig. 5). The selected method thus influences all dependent tree variables. For that reason, these variables are automatically modified according to the actual tree position. Still, it is important to keep in mind, which method of the tree position calculation is being used for the particular trees. Both methods may be calculated with or without the terrain cloud. This is used for adjustment of the tree base Z coordinate. It is strongly recommended to use terrain adjustment if the terrain cloud is available. In such a case, the influence of the tree cloud lowest outliers on the resulted tree base position is lowered (Fig. 5). This is particularly important on steeper slopes. The possibility to calculate tree position without terrain is thus set aside for cases when terrain cloud is missing.

Tree Base Position by Lowest Point

The tree base position computed by the lowest point method can be used via *Trees → Position lowest point*. The dialog window enables the selection of trees whose base positions should be calculated. Besides, it is necessary to set the vertical distance [cm] from the lowest point of the trees to set the tree points included in the calculation. The terrain cloud, if available, should be selected to adjust the Z coordinate to the terrain level. Correspondingly, the number of the closest terrain points (N) employed in this adjustment has to be set. The XY position is then computed as a median coordinate of the points of the tree that are lying between the lowest point of the tree cloud and the user-defined distance. The Z coordinate is defined as the median Z value of N closest points of the terrain at the XY position. If there is no terrain cloud available, the Z value is defined by the lowest tree point. The tree base position is displayed as a sphere with the center at the position and radius of 5 cm.

Tree Base Position by RHT

The tree base position computed by randomized Hough transformation (RHT) is accessed via *Trees → Position RHT*. The dialog window enables users to select the tree(s) for tree base position calculation and set the number of RHT iterations. It is recommended to use at least 200 iteration steps. The terrain cloud, if available, should be specified to adjust the Z coordinate to terrain level as well as the number of the closest terrain points (N) used for that adjustment should be set. The iteration takes the lowest point of the tree cloud as an initial position. The tree base position by RHT then computes the tree base position using centers of two circles fitted by RHT to the stem at 1.3 and 0.65 m above the initial or previously computed position. The RHT position is then defined as the intersection of the vector aiming from the upper circle center (at 1.3 m) to the lower circle center (at 0.65 m) and the horizontal plane defined by the median Z value of the N closest points of the terrain cloud.

Diameter at Breast Height (DBH)

The diameter at breast height (DBH) is computed from the subset of points of the tree cloud (so-called DBH cloud). This cloud lies between 1.25 m and 1.35 m above the tree base position. It is highly recommended to build the DBH cloud on the tree base position adjusted to terrain level as mentioned in section 7.3, otherwise, the DBH might be significantly misrepresented. There are two methods of DBH calculation implemented in the 3D Forest: i) based on the randomized Hough transformation (RHT) and ii) based on the least square regression (LSR). Both methods use the same DBH cloud as input, however, their results may differ. The RHT is a robust method if there are enough points of the segmented stem. This method is however computationally more demanding. On the other hand, the LSR function works well even with a small number of points of the segmented stem. However, it frequently overestimates the diameter value as it is very sensitive to the presence of outlying points. A significant difference between results given by these two methods points to the low quality of the tree segmentation or the overhanging branches. This may be fixed by the Tree cloud edit function (8.1) or by the DBH cloud edit function (8.2), where only the points of the tree subset from 1.25 to 1.35 m above the ground are adjusted.

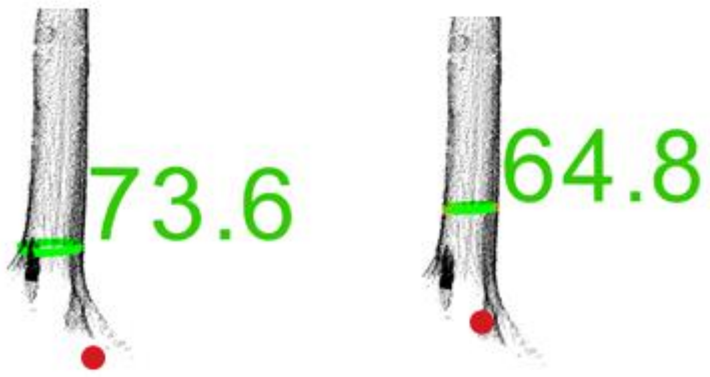


Figure 5: Differences between the DBH computed according to the tree base position estimated from the tree's lowest points (left) and the DBH recomputed according to the tree base position adjusted to the terrain cloud (right). The tree base position is represented by the red dot.

DBH by Randomized Hough Transformation

Computing DBH using randomized Hough transformation is done via *Trees* -> *DBH RHT*. The dialog window gives the option to select trees of interest and set the number of iterations. As always, the amount of iterations is a trade-off between computation time and accuracy. It is recommended to use at least 200 iterations for fast computation. A higher amount of iterations costs more computational time with rising accuracy. The use of 2000 iterations already provides fairly consistent results. Resulting DBH is displayed as a 10 cm high cylinder with the diameter of the best-fitted circle.

The method itself is based on the parametric description of objects within the polar coordinate system. The DBH subset of the tree cloud (i.e. from 1.25 to 1.35 m) is projected to a horizontal plane. The Z coordinates are transformed to 1.3m. Then, the scheme searches every possible center of the circle for each point of the subset. The most frequent circle center is selected as a result (Xu and Oja, 1993).

DBH by Least Squares Regression

Computing DBH using Least squares regression is done via *Trees* → *DBH LSR*. The dialog window gives the option to select trees of interest. Resulting DBH is again displayed as a 10 cm high cylinder with the center and the diameter of the best-fitted circle. As already mentioned, the LSR method is very sensitive to outlying points that do not represent the tree stem (Fig.6). Therefore, this method is suitable for tree clouds with clearly visible stems.

The method starts similarly as DBH RHT, i.e. DBH subset of points lying between 1.25 and 1.35 m is projected to a horizontal plane and the Z coordinates are transformed to 1.3m. Then the circle is fitted to these points by the Least-squares regression. The LSR method is based on minimizing the mean square distance between the fitted circle and data points. The circle fitting is done employing the Gauss-Newton method (Chernov and Lesort, 2003).

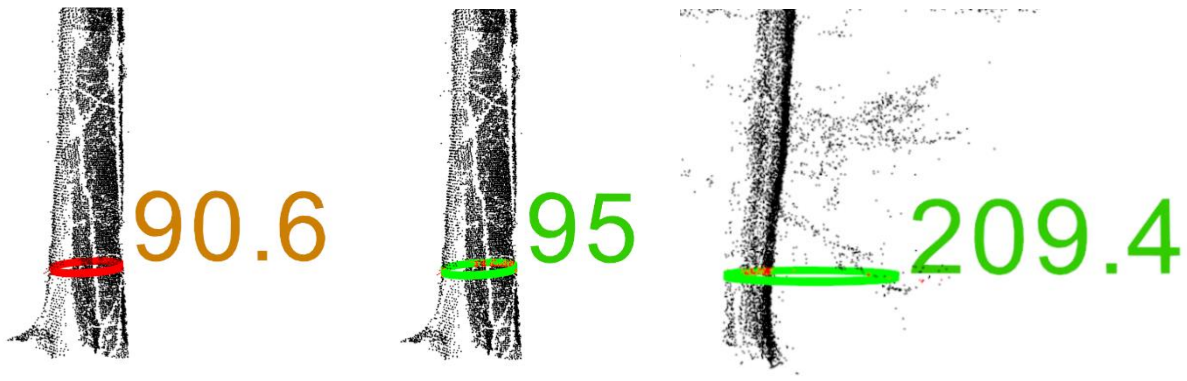


Figure 6: Differences between the RHT (left) and the LSR (middle and right) method; the wrong value given by LSR is caused by an overhanging branch, which was included in an automatically defined DBH subset of the tree cloud.

Tree Height

To obtain the tree height go to *Trees → Height*. There is an option to select trees of interest in the dialog window. The result is displayed in meters on the top of the tree. This tool also displays a vertical line from the tree base position to the highest point.

Tree Length

The tree or cloud length calculation is done via *Trees → Length*. The dialog window gives an option to select tree clouds of interest. The tool calculates Euclidean distance between the most distant points in the tree cloud and displays their connection. The cloud length is displayed in meters at the bottom of the tree. This tool is suitable for the calculation of the real length of inclined or lying trees.

Convex Planar Projection

The tree convex planar projection is evaluated via *Trees → Convex planar projection*. The dialog window gives an option to select tree clouds of interest. The convex planar projection tool calculates and displays the area of the convex hull of the tree cloud orthogonally projected to the horizontal plane at the tree base height. The result can be exported/imported to the ArcGIS polygon shapefile (see Exporting tree planar projection in section Data Export.

The calculation is based on the Gift wrapping algorithm (Rosén et al., 2014). As the first step, the point A with the lowest *y* coordinate is found. This point (A) is the first and the last point of the polygon. Next, two vectors are defined. They are the vector defined by points A and (-1,-1), and the vector defined by point (A) and another point of the tree cloud (X). The angle between these vectors (α) is then calculated for each point (X) of the point cloud. The point (X) that leads to the largest angle (α) is then selected as the second point in the polygon. Other points of the polygon are added similarly, only the vector defined by points A and (-1,-1) is replaced by a vector given by the last two points of the polygon. This continues until the last point is equal to the first point A.

Concave Planar Projection

The tree concave planar projection can be computed via *Trees → Concave Planar Projection*. The tree cloud of interest is selected in the dialog window as well as the maximum edge length, i.e. the threshold distance TD in cm. The tool calculates and displays the area of orthogonal concave tree projection to the horizontal plane in the tree base height and the result can be exported/imported to the ArcGIS polygon shapefile (see 9.2).

The calculation is based on the Gift wrapping algorithm and the modified Divide and Conquer algorithm (Rosén et al., 2014). In the first step, the convex hull is computed similarly as in the previous section. Then edges longer than the initial threshold/searching distance (TD) are found. These edges are split in the next step and the algorithm looking for points inside the polygon is as follows: A, B are adjacent points of the convex hull and X is a point tested to be added in between them. If the distance AX is shorter then TD and BX < AB, or BX < TD and AX < AB, then X is considered as a possible candidate. The edges created by split must be shorter than the original one. If more than one point complies with these rules, the one with the largest α AXB is selected. These conditions avoid too sharp angles in the new polygon.

Once the new point X was selected, the algorithm continues with edge AX or BX if one of them is longer than TD. In the case that edges may not be split, because there are no points to satisfy the split conditions, the algorithm increases TD by 10 cm and repeats the split procedure. This continues until all edges are shorter than TD.

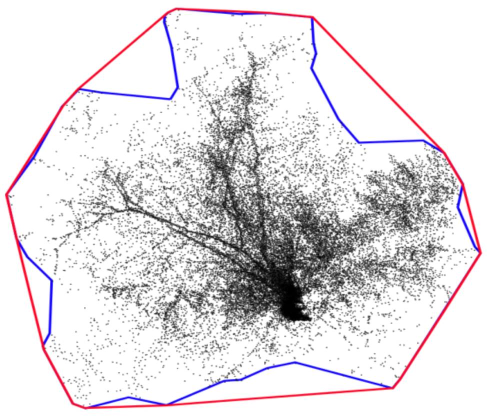


Figure 7: Planar projection of the same tree, convex (red), concave (blue).

Stem Curve

The stem diameters within 1 m intervals along the tree length can be received via *Trees → Stem Curvature*. The stem diameters are computed as circles by Randomized Hough transformation (see DBH by randomized Hough transformation) from 7 cm high slices of the tree cloud. They are displayed as 7 cm high cylinders defined by the RHT fitted circles. The number of RHT iterations is set by the user. The algorithm starts with the stem diameter at 0.65 m above the ground, then at 1.3 m and 2 m above the ground. Then it continues computing diameters with 1 m spacing until the new diameter is two times wider than the previous two diameters, i.e. the calculation is terminated when branches forming the crown got involved in the calculation. Because numerous circles are fitted to each tree, the higher number of RHT iterations significantly increases the computational time. Though, for higher accuracy more iterations are recommended. The output of the stem curve calculation is x,y,z coordinates of the centers of fitted circles and the diameters itself. See also Export Stem Curve in the chapter Data Export.

10. Crowns

The crown attributes (Fig. 8) can be evaluated if the tree position and tree height are computed. Crown height, crown base height, crown width, crown centroid, and crown position deviation are then computed automatically. These variables are recomputed automatically if the tree position is changed.

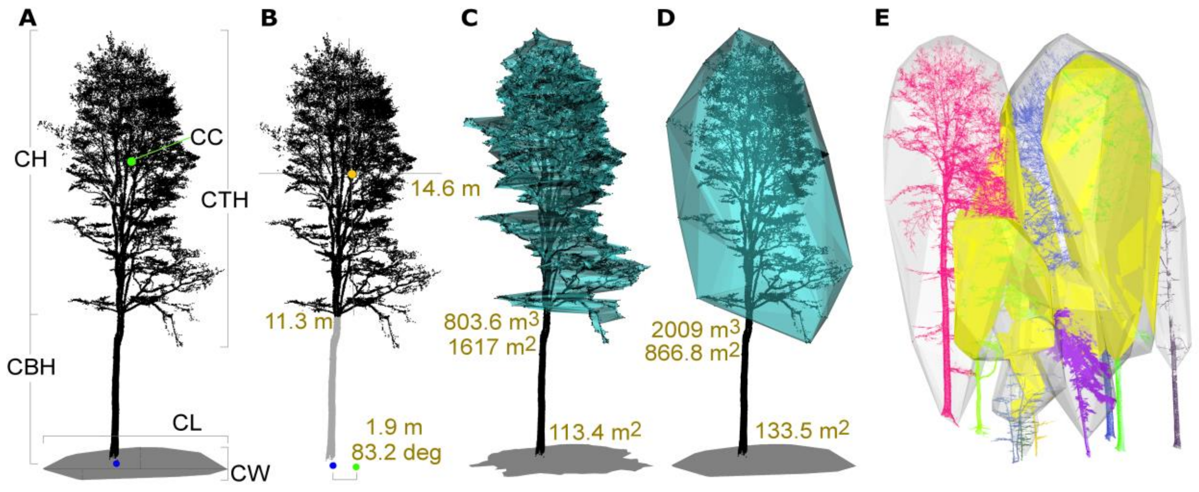


Figure 8: A - The description of the crown attributes: crown height (CH), crown base height (CBH), crown center (CC), crown total height (CTH), crown length (CL), crown width (CW). **B** – The example of calculated parameters and the deviation of the projection of the crown center (green) from the base of the trunk (blue). **C** – The concave planar projections and volume/surface of the crown as calculated according to the concave polyhedron. **D** – The convex planar projections and crown volume/surface calculated using a convex envelope. **E** – The example of the visual output of the 3D Forest when calculating the shared space of crowns (yellow) using 3D convex covers.

Crown Cloud Adjustment

The crown cloud can be adjusted manually via *Crowns → Set manual* or automatically via *Crowns → Set automatic*. The dialog window gives an option to select the tree of interest.

The manual mode then enables to remove the stem points.

The automatic mode creates cross-sections of the tree along the z-axis with the height 0.5 m at first. The width of each section is computed by averaging the x and y-axis values of the respective section. The widths are successively compared from the lowest to the highest section. If three consecutive sections are wider more than 25 % than the last thin section, the last thin section is marked as a starting place for detailed search. The detailed search begins by setting a subset of points with a height of 10 cm at the beginning of the thin section. Using LSR the stem diameters at the start and the end of the subset of points are computed. These diameters enable to predict the center of the next diameter, located another 10 cm along the section. The diameter itself is then computed using only the points lying within the range of the doubled value of the preceding diameter. This is to avoid the influence of overhanging branches. Predicting the new diameter center and diameter evaluation continues until the difference between the last two diameters is less than 25%. This point is then considered as the base of the crown.

Crown Volume by Voxels

The crown volume by voxels is estimated via *Crowns → Volume by voxels*. The dialogue window enables to select the tree and to set voxel size in cm. The volume is shown in the project attribute table. Voxels are not visualized due to the computational demands.

Crown Volume and Surface by Concave Polyhedron

The crown volume and surface area by cross-sections are computed via *Crowns → Volume and surface by concave polyhedron*. The dialogue window enables to select the tree of interest, set the section height, and set the threshold distance for section concave hull. In case that other than implicit values are set, the crown position and position deviation are recomputed with external points extracted by these new parameters. The crown position is calculated as an average coordinate from external points, i.e. the 2D concave hulls of horizontal cross-sections (slices) with the threshold distance 1 m and section height 1 m; see Concave Planar Projection. The external points can be visualized by the icon . Crown volume is then computed as the total amount of sections volume, where each section volume is computed by its planar projection area (created by concave hull) times section height. The surface area is computed by strip triangulation. The algorithm is reliable, although there are cases when intersecting triangles may occur. To show/hide resulting objects use icon . The triangulation algorithm connects vertices of concave hulls of adjacent sections by the shortest possible sum of edges.

Volume and surface by the 3D convex hull

The crown volume and surface area by 3D convex hull are accessible via *Crowns → Compute 3D ConvexHull*. The computation (3D Delaunay triangulation and mesh parameter algorithms) is done using the VTK libraries. The dialogue window allows selecting the tree of interest and offers the checkbox for computation using all crown points. The external points and points belonging to the lowest and highest sections are used by default. This provides a significant decrease in computational time with comparable results as if all crown points are used. To show/hide resulting shapes use icon .

Crown intersections

The crown intersections evaluation is available via *Crowns → Intersections*. All existing crowns are tested for intersections. The existing intersection between two crowns is computed using VTK as Boolean AND in 3D space. The convex shapes are only possible for the evaluation thus to create the 3D objects volume and surface by 3D convex hull have to be computed first. The following variables of intersecting crowns are computed: horizontal angle (azimuth), vertical angle (from horizontal plane), and distance from crown position to intersection center of gravity, intersection volume, and surface area. These attributes are listed in the intersections table, called by icon . Intersecting parts can be shown/hidden by icon , the lines connecting crowns positions with intersection center of gravity, and angles are also visible.

11. QSM Models and Tree assortments

The *Quantitative Structure Models (QSM)* menu offers methods to obtain information about branches and stem (volume, length, branching order) and methods for tree analysis (other quantitative parameters of trees). The QSM models may be employed if a tree position and height are known.

Tree Reconstruction

Tree reconstruction is the first step of the QSM analysis.

The input parameters are the tree of interest, voxel size, and multiplicator that defines the close neighbor, i.e. maximal distance of used voxels. The reconstruction starts by dividing the whole tree into voxels. The points that are inside the voxel, as well as points and voxels that are close neighbor (voxel size * multiplicator), are recognized for each voxel. The stem reconstruction starts by neighbor search at the lowest voxel. All neighbors connected in one piece into the homogeneous part are considered as segments. If neighbors create two or more groups of non-connected voxels they are classified assible branches. The segment is closed at that place and the analysis continues up to the tree length. Finally, the information about voxels, ordering, i.e. parent, and child succession of segments, is known for each segment. Once the whole tree is divided into the segments, the tree reconstruction itself follows. Firstly all segments without children are said to be the ending segments. Parents of each of the ending segments are followed to the lowest voxel and their lengths are computed. The stem is the longest sequence of segments. These segments are merged into a single segment and all its children are assigned. The reconstruction of branches of the first order follows with similar semantics until it reaches the stem segment. After all first-order branches are reconstructed, branches of second, third, etc. order follow until all segments are assigned to the tree.

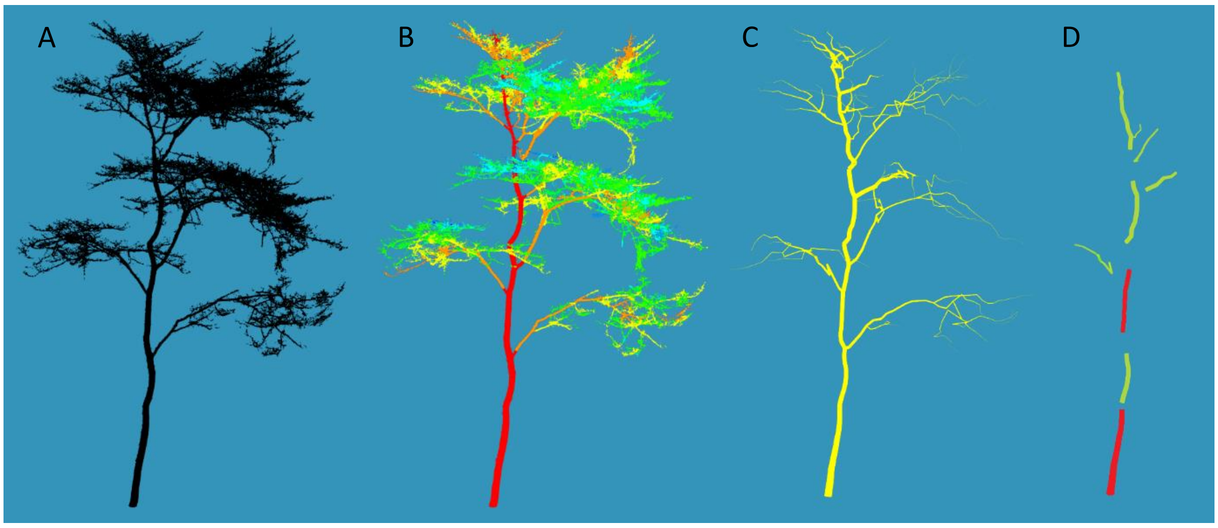


Figure 9: A – unprocessed tree point cloud, B – results of tree reconstruction: stem (red) and branches (green to blue according to branching order), C – cylindrical model of a tree, D – wood assortments.

QSM model

After the tree reconstruction, the parametric model of the tree can be evaluated. First of all the tree of interest has to be specified. Since the cylinder estimation is done by randomized hough transformation (as for DBH) the number of iterations has to be set then. The size of the estimated cylinders defines the height of the cylinders used for reconstruction. Reconstruction can be further restricted to the branches up to given order, given length, or given diameter. The last option is the stem profile evaluation checkbox. This result can differ from those given by the algorithms offered in the “tree menu”. The final results of the QSM reconstruction appear as connected cylinders for each branch with an attribute table containing information about volume, length, and order if each fitted cylinder.

Tree assortments

The final step is to estimate possible wood assortments (as the base for timber value estimates) according to Czech industry standards for wood assortments. The only input is the trees of interest. The evaluated attributes of the tree are convergence, skewness, diameter, length, and number of connected branches. The algorithm classifies the whole tree represented by cylinders into quality classes, higher quality classes are prioritized.

Table 2: Example of tree criterium for the classification of wood quality.

Class	Convergence	Skewness	Diameter size	Length	Branches
1.	up to 1 cm	1,5 cm/m	45 cm	3 m	absent
2.	up to 1 cm	2 cm/m	28 cm	3 m	up to 3cm
3.		3 cm/m	20 cm	2,5 m	up to 4 cm
4.		6 cm/m	7 cm	2 m	
5.		10 cm/m	7 cm	1 m	
6.					

12. Data Export

All data exported from the 3D Forest are re-transformed into their original coordinates system.

Clouds Export

The cloud data are exported via *Project → Export*. There is an option to choose an output format (.txt, .ply or .pcd). Then the cloud for export, the name, and the location of the new file are selected.

Tree/Crown Parameters Export

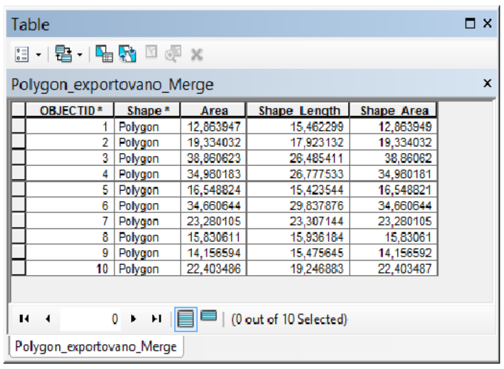
All calculated tree/crown parameters as a tabular output can be exported via *Trees → Export tree parameters* or in case of the crown via *Crown → Export crown parameters*. The name, the location, and the separator within the exported .txt file have to be specified. The parameters that were not evaluated are exported as -1 value.

Tree Planar Projection Export

The planar projections of trees can be exported via *Trees → Export convex planar projection* or *Trees → Export concave planar projection*. The name and the location of the new file have to be selected. The planar projection is exported in one file for all trees. The projections that were´t computed previously during the ongoing 3D forest session are not exported. Both functions create .txt file with coordinates of vertices of the polygon in the following format: Tree ID;Xstart;Ystart;Xend;Yend;Z

Tips: the data import into ArcGIS and polygon or polyline shapefiles creation

Add an exported file to your project and use function XY TO LINE in ARCTOOLBOX → DATA MANAGEMENT TOOLS → FEATURES. This creates a new layer containing geodetic line features constructed based on input files. To create a closed polyline from this layer use function UNSPLIT LINE in ARCTOOLBOX → DATA MANAGEMENT TOOLS → FEATURES. Afterward, it is possible to create polygons that are not overlapping to use function FEATURE TO POLYGON in ARCTOOLBOX → DATA MANAGEMENT TOOLS → FEATURES. To create overlapping polygons it is possible to use function CONVERT POLYLINES TO POLYGONS from „Tools for Graphics and Shapes“ that is freely available at http://www.jennessent.com/.



Table

Polygon_exportovano_Merge

OBJECTID *	Shape *	Area	Shape_Length	Shape_Area
1	Polygon	12,863947	15,462299	12,863949
2	Polygon	19,334032	17,923132	19,334032
3	Polygon	38,860623	26,485411	38,86062
4	Polygon	34,980183	26,777533	34,980181
5	Polygon	16,548824	15,423544	16,548821
6	Polygon	34,660644	29,837876	34,660644
7	Polygon	23,280105	23,307144	23,280105
8	Polygon	15,830611	15,936184	15,83061
9	Polygon	14,156594	15,475645	14,156592
10	Polygon	22,403486	19,246883	22,403487

(0 out of 10 Selected)

Polygon_exportovano_Merge

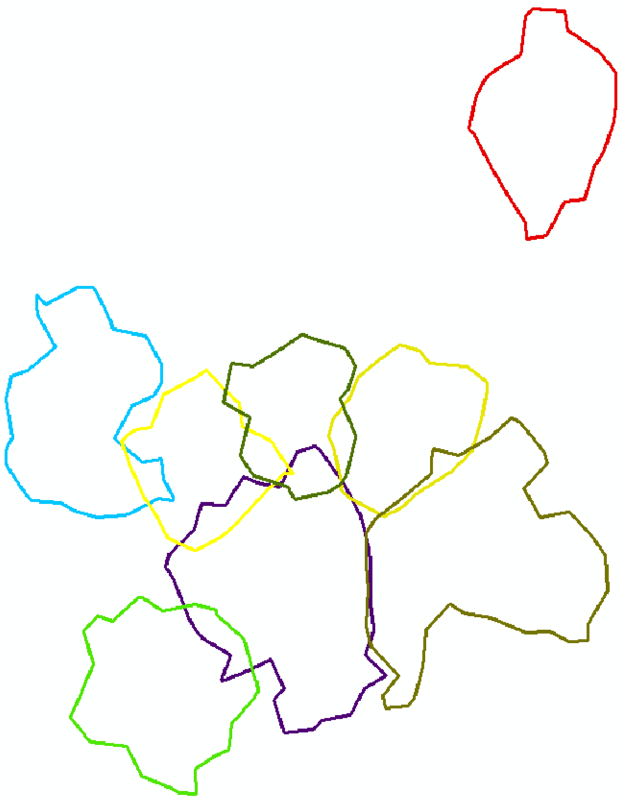


Figure 10: Polygons of the concave tree planar projections as imported in the ArcMap.

Stem Curve Export

The stem curve can be exported via *Trees → Export Stem Curve*. The dialogue window allows specifying the tree of interest, the file name, and the destination folder. The exported file is in the .txt format with the following structure:

Tree name diameter: 70.6 67.6 64.6 ..... -1 -1 -1
Tree name x: 182.474 182.5 182.455 ..... -1 -1 -1
Tree name y: -1594.55 -1594.56 -1594.57 ..... -1 -1 -1
Tree name z: 1017.6 1018.25 1018.95 ..... -1 -1 -1

Diameters and coordinates of centers of fitted circles are in ascending order of tree height, i.e. from 0.65 to 1.3, 2, etc. The value -1 indicates data that were not evaluated.

Intersections Parameters Export

The parameters of intersections can be exported via *Crowns → Export intersections parameters*. The dialogue window facilitates to set the name of the new file and the destination folder. Parameters are exported as .txt file with the next structure: the tree name, the name of the intersecting tree, the intersection volume (m^3), the surface area of the intersection (m^2), the horizontal angle (°), the vertical angle (°), and the distance (m). The semicolon is used as a separator.

13. Other Features

This menu comprises functions that are generally applicable to all cloud types.

Cloud Merge

This function helps to join two point clouds into a new one. It can be called via *Other features → Cloud merge*. In the dialog window select the clouds to be merged, set the name of the new cloud, select its type (e.g. Tree, Base, etc.), and check the box if you want to delete the input clouds.

Cloud Subtraction

The cloud subtraction is done via *Other features → Cloud subtraction*. The dialog window offers the option to select the point clouds to be subtracted and set the name of the output cloud. The tool subtracts the smaller cloud from the bigger cloud. Identical points of both clouds are removed from the bigger cloud and the result is saved as a new cloud.

Voxelize Cloud

The reduction of the point cloud density is possible via *Other features → Voxelize cloud*. The dialog window gives the option to select the cloud of interest, set the name of the voxelized point cloud, and the size of the voxels in cm. The tool generates a voxelized point cloud from the input by centroids of voxels that included at least one point of the original point cloud.

Create Convex or Concave 2D Hull

The convex/concave hull of any cloud can be created via *Other features → Create convex hull or Create concave hull*. These functions evaluate the convex/concave hull of the orthogonal projection of the cloud into the horizontal plane at the height of the lowest point and save it as a new cloud (type Others). The same algorithms as described in the chapters Convex planar projection and Concave planar projection are used. The tool thus creates a layer with the same extent and boundaries as the area in 3D Forest. This may be further used to transfer areas of interest from the 3D Forest to a GIS. The hull cloud should be exported as .txt if imported into ArcMap for polyline or polygon creation, for more details see chapter Data Export.

Remove duplicate points

The redundant points may be removed via *Other features → Remove duplicate points*. The dialog window enables to select the cloud for filtering. The point cloud is divided into smaller point clouds using octree search (octree resolution is 20 cm) and each cloud is separately voxelized with 1 mm voxel size. After that, the clouds are merged back.

Save into tiff

The current view from the visualization area can be saved via *Other features → Save into tiff*. Then set the name for the .tiff file and the destination folder.

Change Background-color

The background color of the viewer can be changed via *Other features → Change background color*. Then select a new color.

14 References

Chernov, N. and C. Lesort (2003). Least squares fitting of circles and lines. arXiv preprint cs/0301001.

McDonald, J. (2014). The Hough Transform-Explained and Extended. [online] cited 17.7.2014. Available at: www.cis.rit.edu/class/simg782.old/talkHough/HoughLecCircles.html.

Rosén, E., E Jansson and M. Brundin (2014). Implementation of a fast and efficient concave hull algorithm. Available at: http://www.it.uu.se/edu/course/homepage/projektTDB/ht13/project10/Project-10-report.pdf. Project Report. Uppsala University.

Rusu, R. B., S. Cousins and Ieee (2011). 3D is here: Point Cloud Library (PCL). 2011 Ieee International Conference on Robotics and Automation (Icra).

Xu, L. and E. Oja (1993). Randomized Hough transform (RHT): basic mechanisms, algorithms, and computational complexities. CVGIP: Image understanding **57**(2): 131-154.

Statisticall outlier removal filter:
https://pcl.readthedocs.io/projects/tutorials/en/latest/statistical_outlier.html#statistical-outlier-removal

Radius outlier removal filter:
https://pcl.readthedocs.io/projects/tutorials/en/latest/remove_outliers.html#remove-outliers