**Forest-Taxator**

**URL:** [GitHub - maciej-malaszek/forest-taxator](https://github.com/maciej-malaszek/forest-taxator)

**Date:** 11 October 2021

**Author:** Maciej Malaszek

**Version:** 1.0.0

**Description:** This software is developed for the terrestrial point cloud data processing. The two main functionality of this tool is (1) tree detection in the point cloud, (2) development of three-dimensional models for individual trees.

**Installation:** The software is written in C++ and runs on .NET Core platform. So, download Microsoft visual studio because it has all the dependencies which this tool will require.

If the .NET is there, it should look like:

H:\>dotnet

Usage: dotnet [options]

Usage: dotnet [path-to-application]

Options:

-h|--help Display help.

--info Display .NET information.

--list-sdks Display the installed SDKs.

--list-runtimes Display the installed runtimes.

path-to-application:

The path to an application .dll file to execute.

Then, you need to change the directory where you have saved you zip file of this tool and give the path of the main folder of forest-taxator like here:

cd C:\forest-taxator

once you changed the directory and given the correct path, simply put the below command:

C:\forest-taxator>dotnet build -c Release forest-taxator

It should work and will look like in the attachment and then just follow the commands for the analysis of the point cloud.

Some illustrative examples:

There are functions like:

analyze – allows to aggregates all necessary function to complete the study step by step

convert – conversion of file format into .xyz file.

approximate-trees – approximate tree trunk using ellipses

detect-trees – detect trees

filter

slice

terrain

tree-height

help – display more information on a specific command

version- Display version information

1. Extract terrain height map from example.xyz into terrain.thf.

**./ForestTaxator.Application analyze example.xyz -o output/terrain**

(replace -o to the output file location, add location of example.xyz)

1. Extract tree height map into tree-height.

**./ForestTaxator.Application analyze tree-height example.xyz -o output/terrain**

1. Create the sliced point cloud

**./ForestTaxator.Application analyze slice example.xyz -t output/terrain -o output/sliced**

-t/terrain -> path to terrain heightmap file (use to normalize height of each point in cloud)

1. Create small groups of point based on their neighbourhood

**./ForestTaxator.Application analyze filter output/sliced -c Configs/FiltersConfiguration.json -o output/filtered.gpd**

Input: sliced point cloud from slice step; Terrain height map

-c -> path to FiltersConfiguration.json file

-o -> path & filename where output file has to be saved

1. Convert gdp file to XYZ file.

**./ForestTaxator.Application convert -i GPD -o XYZ output/filtered.gpd output/filtered.xyz**

1. Create a collection of point groups. Collections will be create based on XY location.

**./ForestTaxator.Application analyze detect-trees output/filtered.gpd -o output/detected –export-preview**

Input: GPD file from filter step of collection of XYZ files from detect-trees step

--expert-preview -> for each exported GPD file, XYZ file is created so it can be viewed in 3d cloud viewer

-o -> path to Directory where output file has to be saved

1. Create approximation of tree trunk using genetic algorithm.

**./ForestTaxator.Application analyze approximate-trees output/detected -c Configs/ApproximationConfiguration.json -o output/approx. -t output/tree-height –export-preview**

Input: GPD file from detect-trees step; Tree height heightmap

--export-preview -> exports collection of approximating ellipses into XYZ file. 4th value in point represents error equal to mean distance ellipse border and cloud point

--smooth – smooth tree using regression on all nodes. It replaces ellipses with circles

-h / --node-height ->float number representing height of single slice. Required if file was sliced with different height than default 0.1 m

-t -> tree-height map -> path to tree height map file

-o -> path to Directory where output file must be saved