Politecnico di Torino
Department of Environment, Land and Infrastructure Engineering

26 Luglio 2022

EBE Example Walkthrough

This example provides an overview of the EBE procedure to map riparian biomass from georeferenced point clouds. For further details, please refer to the following article:

Latella, M., Raimondo, T., Belcore, L., Salerno, L., and Camporeale, C. (2022), On the integration of LiDAR and field data for riparian biomass estimation, *Journal of Environmental Management*

Premise:

- In this example, we apply EBE to a small reach of the Orco river, in Northwest Italy. The results are illustrative and not intended to be comprehensive.
- The example is completely run from the Matlab Editor (i.e., no GUI is needed/provided).
- The following description refers to the headings that can be found in the code.

parameter setup

The parameters in this section must be tuned according to the features of the used computer, the input data and the field measurements.

input and output definition

The names in this section must be modified according to the directory and filenames.

parallel definition

Parallelization is not mandatory but is strongly recommended, especially for large study areas. The script is provided using parallelization. If you want to turn off this option, you can comment this section and rewrite all the *"parfor"* in *"for"* throughout the script.

directory creation

The code creates the directories where to save the EBE's outcomes.

input retrieval

input

The code retrieves the input files based on the previously given names and directories.

first operations

The code performs some operations over the input files. For instance, if the shapefiles are multipolygon, it subdivides them into individual polygons.

biomass computation

output initialization

The code initializes the json files of the biomass and the associated errors.

loop over the file list

[The operations described in the following are repeated for each input file.]

The code imports the RGB file and the shapefile describing the wet channels, the region of interest (RoI), and the shrub area (SA). Then, it defines the masks indicating where to look for shrubs.

First, it determines the individual masks associated with the RoI (Figure 1) and the three color bands (Figures 2, 3 and 4).

Figure 1: RoI mask.

Figure 2: Red mask.

Figure 3: Green mask.

Figure 4: Blue mask.

Second, it intersects the color masks generating an RGB mask (Figure 5), and intersects the RGB and RoI masks again to produce the first version of the vegetation mask (Figure 6).

Figure 5: RGB mask (i.e., the intersection among the red, green, and blue masks).

Figure 6: Vegetation mask, first version (i.e., the intersection between the RoI and the RGB masks).

Third, it generates a mask for each water polygon and removes them from the vegetation mask (Figure 7). This step is necessary since identifying vegetated areas based on the RGB might misclassify water as vegetation.

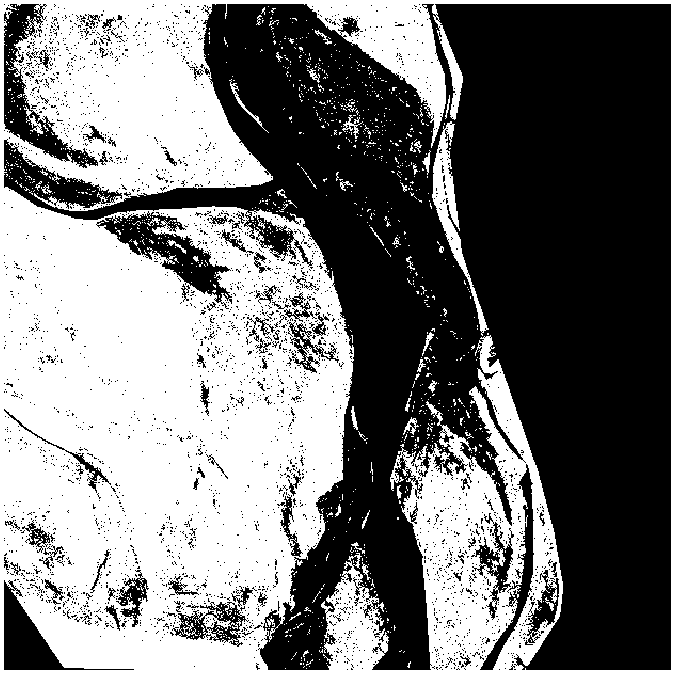

Figure 7: Vegetation mask, second version (i.e., after removing the water mask(s)).

Since the vegetation mask includes both trees and shrubs, the code removes all the islands with standing trees as indicated by the input files (Figure 8).

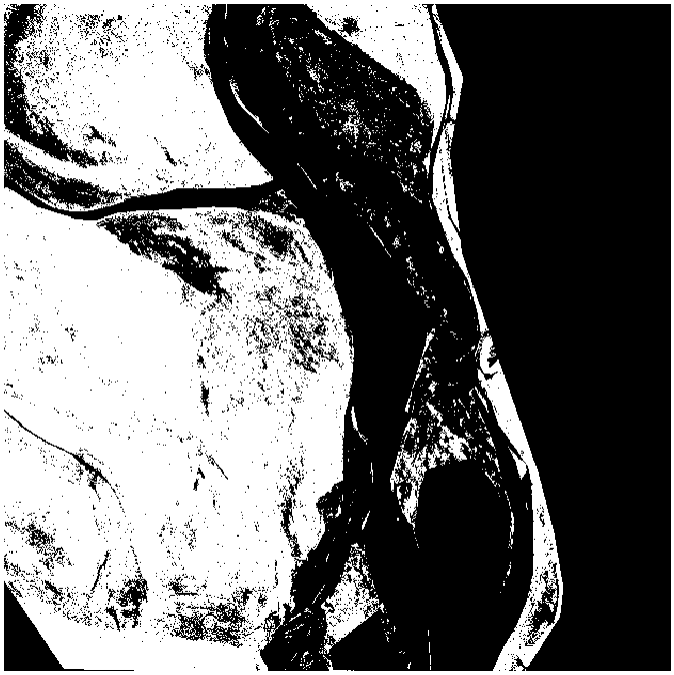

Figure 8: Vegetation mask, third version (i.e., after removing the island mask(s)).

At this point, the code starts computing the biomass of the trees inside the SA. Once the biomass of a tree is computed, it removes its projected area from the vegetation mask. The same procedure is repeated for the trees on islands or outside the SA but still in the RoI. In the end, the vegetation

mask only represents shrubs (Figure 9, left), and the code can identify the shrub clusters working on the mask (Figure 9, right).

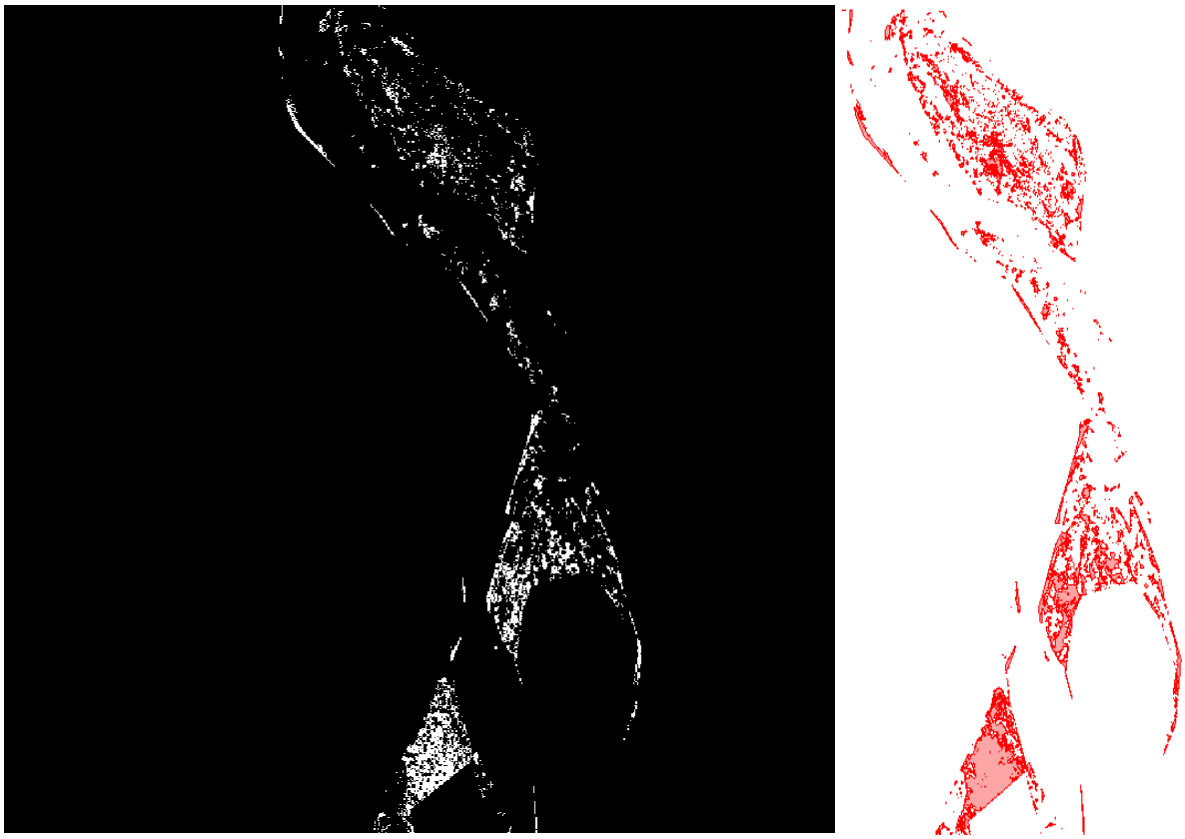

Figure 9: Vegetation mask, fourth version (i.e., after removing all the trees) on the left. The identified shrub clusters on the right.

The code separates the clusters and works on them to compute their biomass. For this purpose, it clips the input point cloud according to the individual clusters and calls the auxiliary function *EBE_concave_hull.m*. This function reconstructs the volume of the shrubs. Then, volumes are converted into biomass.

output generation

EBE output consists of json files listing the biomass (or the error) of every tree and shrub in the RoI. They can be visualized in GIS environments (Figure 10) and processed to obtain maps.

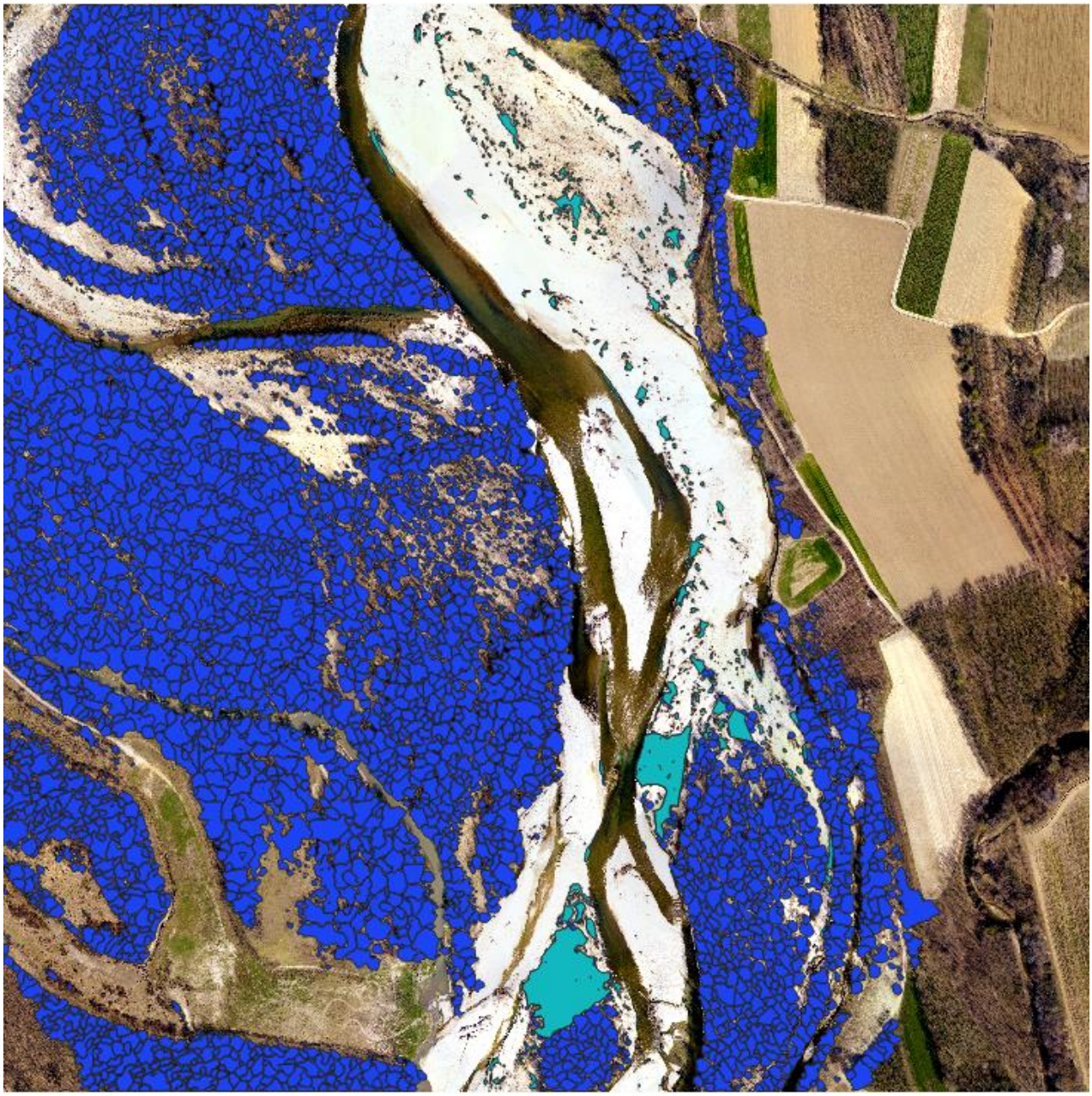

Figure 10: EBE output visualized in qGIS. Blue and cyan indicate trees and shrubs, respectively.