Validating the CENITH R-package as a new approach for tree segmentation of alpine treelines

Jan Schwalb[1] and Niklas Werner[2]

[1] *M.Sc. Physical Geography, Philipps-University Marburg, Matr.-No.: 2452600, Schwalbj@students.uni-marburg.de*
[2] *B.Sc. Geography, Philipps-University Marburg, Matr.-No.: 3218235, Wernern@students.uni-marburg.de*
Lecturer: Prof. Dr. Maaike Bader

November 12, 2020

The investigation of patterns of alpine treelines is everpresent in alpine biogeographical research. This year, 2020, the new R-package CENITH was introduced to make tree segmentation easier and accessible for novice users. In this study, the CENITH package was tested against established approaches and evaluated according to its applicability and performance. CENITH could yield convincing results especially for the segmentation of trees, while its performance to segment shrub patterns was slightly inferior to some of the established methods.

1 Introduction

Eversince the beginning of biogeographical research, the conspicuous structure of alpine treelines has been of great interest for researchers all over the world [Slatyer and Noble, 1992]. By comparing the spatial properties of alpine treelines on different continents, mountain ranges or even just differently exposed slopes, one may notice significant differences in the spatial pattering of those treelines [Harsch and Bader, 2011].

With the dawn of satellite remote sensing in the 1960's, large scale investigation on spatial structures in alpine treelines could be conducted [Moore, 1979]. Especially active remote sensing in the form of light detection and ranging (LiDAR) proved valuable. Yielding point cloud data, LiDAR made it easier to segment individual trees [Latifi et al., 2015]. Since this was at first done manually, within the last few decades a rush of automated segmentation approaches flooded the scientific landscape [Guan et al., 2015, Li et al., 2012, Reitberger et al., 2009, Rahman and Gorte, 2009]. Most of the algorithms nowaday used for tree segmentation are based on statistical methods invented for different purposes, such as nearest neighbour approaches [Silva et al., 2016] or growing region algorithms [Dalponte and Coomes, 2016]. Different vegetation zones may require different approaches. So come, that on the one hand, Allen and Walsh mainly used an unsupervised classification approach (not further specified) for their analysis of alpine treelines in the Glacier National Park, Montana [Allen and Walsh, 1996]. On the other hand Wang et al. used a O(r)-ring statistical methods for their study of the treeline patterns of the south-eastern Tibetan Plateau [Wang et al., 2012].

Many of the segmentation approaches mentioned, and even beyond those, are quite error-prone under difficult circumstances [Swetnam and Falk, 2014]. Most of the methods have a hard time dealing with tree shadows, shrubs and krummholz or steep slope inclinations. The scripting language R already provides different packages for LiDAR-based tree segmentation, such as "lidR" [Jean-Romain et al., 2020], "rLiDAR" [Silva et al., 2017] and "uavRst" [Meyer and Reudenbach, 2018]. Those packages do not perform perfectly on all occasions and

their functions are rather hard to grasp for beginners. Therefore, Andreas Schönberg (2020) introduced the CENITH package in R [Schönberg, 2020]. It is supposed to provide an easy-to-use set of functions for parameter tuning and tree segmentation and to perform well, regarding shadows and slope inclination. The purpose of the present study was to use Schönberg's CENITH package from a beginners viewpoint and evaluate its user friendliness, applicability and performance compared to the already established packages lidR, rLiDAR and uavRst. Therefore, the segmentation functions dalponte2016 (lidR), silva2016 (lidR), li2012 (lidR/rLiDAR), chmseg_FT (ForestTools/uavRst) and chmseg_RL (rLiDAR/uavRst) were compared to the TreeSeg function of the CENITH package using manually inserted validation *ground truth*.

2 Materials and Methods

2.1 Study Site

The study area is located in the Austrian region Voralberg, in the northern part of the European Alps. The area was devided into four study sites. They are exposed at the southern slope of Mt. Zimba at approximately 47°05'00.7"N 9°47'47.4"E. The main tree species occuring in the area of interest is the Bog pine (*Pinus mugo*), as well as different species of shrubs. The former often occurs as krummholz. To fit the purpose of investigating treeline patterns, all four study sites are located within the treeline ecotone.

The first site (see figure 1) is dominated by shrubs and lays at the slope with the steepest inclination. The patch of shrubs used for the segmentation approach is the highest vegetational structure in this part of the slope. Study site two lays about 300 m eastwards of the first one. This site is considered to contain more krummholz, as was assumed by manual image supervision. The area used for segmentation purposes is located at the north-western border of the study site. The thinning of the vegetation at this site is more gradual than at the first one, due to a lower inclination. The vegetation at the third site is composed of trees as well as of shrubs, whereas site four merely includes trees. Both sites contain large amounts of bare rock and represent just the upper limits of the treeline ecotone. Study site four includes a patch of trees in the north-eastern corner, which was used for the tree segmentation.

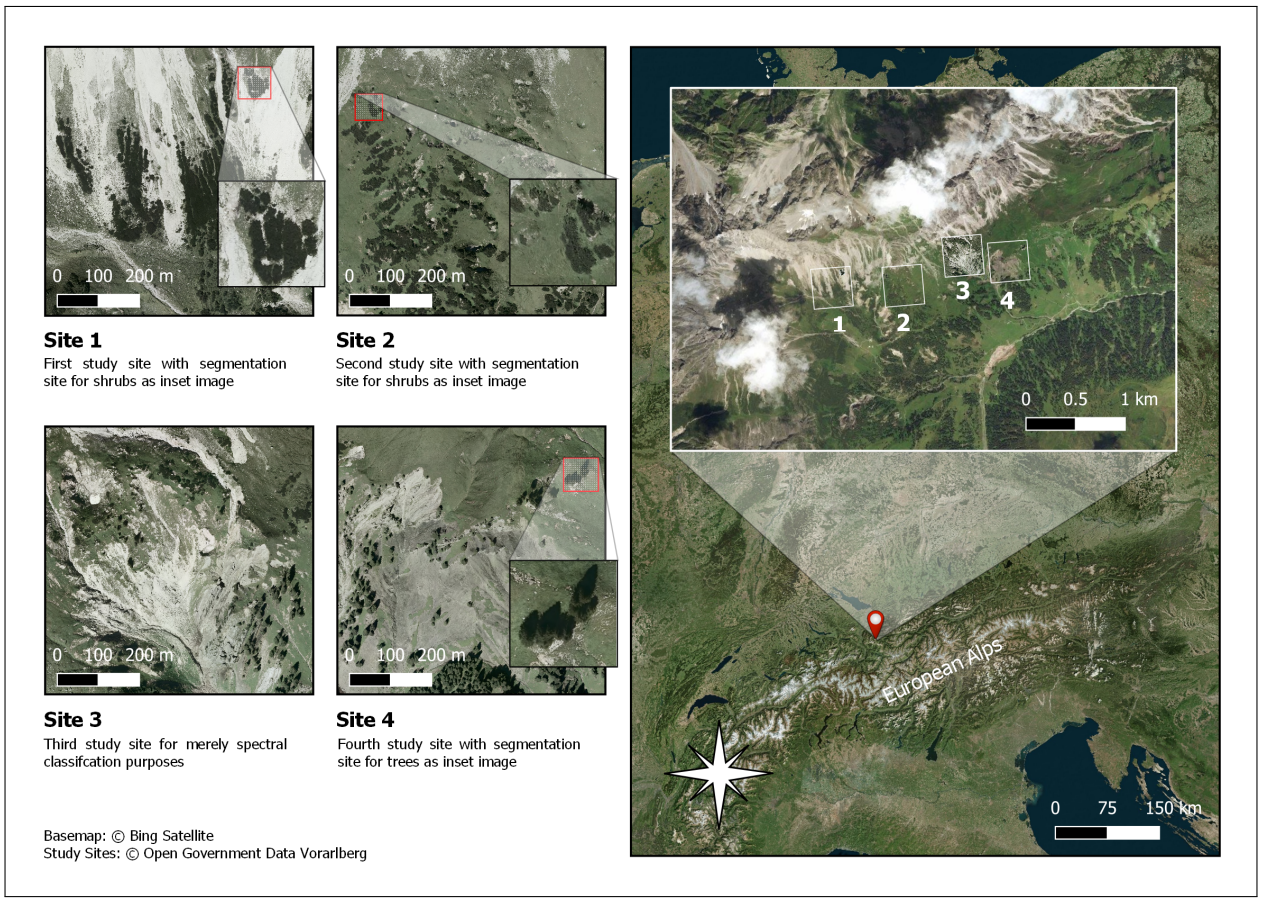

Figure 1: *Location map of the study sites. The RGB images of Voralberg were obtained from vogis.cnv.at and are open source.*

2.2 Data

The main data source, relevant for the segmentation, was high-resolution airborne LiDAR data. The yielded point cloud data was furthermore processed to be partially used as a high-resolution digital elevation and surface models with a spatial resolution of approximately 0.13 m. Further, RGB images were used for the spectral classification. Addditionally, infrared images were used for the spectral classification. Under use of the RGB images and the LiDAR data for treetop identification, validation shapefile were created. Due to the fact, that no physically evaluated ground truth was available, we assume this data as ground truth. The validation points are presented in the Results section for better comparability with the actual results. Deliberately, the points were not plotted inside the segmentation polygons to ensure clarity.

2.3 Preprocessing

Because the LiDAR data contained artefacts, all points higher than 30 m above ground were eliminated. Hence, just trees and shrubs remained. Still, the data contain some errors, which could not be rectified within the margins of this study. We assumed the possible impact of these inaccuracies as minimal.

2.4 Established segmentation algorithms

2.4.1 lidR - Dalponte

The segmentation approach developed by Dalponte and Coomes is based on individual tree crowns (ITC). The ITC's are identified from the LiDAR point cloud by using a region-growing algorithm. For their ITC delineation they adapted a relatively simple method, introduced by Hyyppä et al. [Hyyppa et al., 2001]. This approach is able to find local maxima within a canopy height model (CHM) raster. Those maxima are identified as individual tree tops. Following, it uses a decision tree to "[...] *grow individual crowns around the local maxima.*" [Dalponte and Coomes, 2016]. There are several steps the algorithm goes through: (1) The first one is a low-pass filter that smoothes the rasterized CHM to reduce the number of local maxima. (2) Secondly, a moving window approach is applied to identify local maxima. A local maximum is detected, when a pixel's value is greater than all other values within the window. (3) Then, every local maximum identified is labeled as a initial region around which a "tree crown can grow". This is done by extracting the height of the four neighbouring pixel. They are added to the potential "tree crown" if their vertical distance and height are less some user-defined value. (4) This is done iteratively as long as no more pixels are added. (5) At last, the first-return values from the LAS points are extracted and are turned into polygones by applying a 2D convex hull [Dalponte and Coomes, 2016]. Note, that this approach can be viewed as semi-automatic, since the user has to set the size of the moving window, the small-tree cut-off height and the height difference threshold. Dalponte's algorithm is provided by the function dalponte2016 of the lidR package.

Approach: After loading the CHM and the LAS-files, we segmented the trees using the **segment_trees** function (lidR). Inside this function, the Dalponte's algorithm was used as the individual tree segmentation function. This procedure was repeated for shrubs and tree-shrub-layers (for parameters see table 1). Finally, the results were converted to polygons using the **tree_hulls** function.

	th_tree	th_seed	th_cr	max_cr
trees	2	0.45	0.55	10000
shrubs	0.6	0.45	0.55	10000
tree-shrub	0.6	0.45	0.55	10000
tree-shrub (tree)	2	0.45	0.55	10000
tree-shrub (shrub)	0.6	0.45	0.55	10000

Table 1: *Parameters for the dalponte2016 function*

2.4.2 lidR - Silva

Silva et al. state in their 2016 paper a new and promising approach for individual tree segmentation based on the on the rLiDAR package. For the individual tree detection the FindTreesCHM function is applied to the CHM raster. As in Dalponte's function, it as well uses a local maximum algorithm and a moving window method to detect treetops. For the individual crown delineation it uses the ForestCAS function from the rLiDAR package. This function start by applying a variable radius crown buffer to set limits for the inital crown area. Afterwards, the variable radius is calculated for the individual tree by multiplying the tree height (LiDAR-derived) by a user-set

factor. The merged tree polygons are determined using an area delimitation (see Silva et al. 2016 for more detailed information). The data is then split using a centroidal voronoi tessellation approach [Aurenhammer and Klein, 1999]. This is done to isolate the individual tree polygons. The polygons are then clipped from the CHM [Silva et al., 2016]. The method is provided by the **silva2016** function of the lidR package.

Approach: The approach is the same as for the **dalponte2016** function, since both function operate inside the **segment_tree** function (lidR) (table 2).

	max_cr_factor	exclusion
trees	0.6	0.3
shrubs	0.6	0.3
tree-shrub	0.6	0.3
tree-shrub (tree)	0.6	0.3
tree-shrub (shrub)	0.6	0.3

Table 2: *Parameters for the silva2016 function*

2.4.3 lidR - Li

In 2012 Li et al.developed a new method vor individual tree segmentation based on the relative spacing between trees. Generally, the spacing between the tops of trees ir larger, than the spacing at the base. This approach takes a LiDAR point cloud and identifies the target tree by including point that are nearby, while excluding point further away. To overcome the difficulty of overlaps at the bottom of the tree, the points are classified sequentially, from top to bottom. Hence, points with a larger spacing than the set threshold can be excluded. A minimum spacing rule dictates the spacing for included points. The method is provided by the li2012 function of the lidR package.

Approach: After loading the CHM and the LAS-files, we segmented the trees using the segment_trees function (lidR). Inside this function, the li2012 algorithm was used as the individual tree segmentation function. This procedure was repeated for shrubs and tree-shrub-layers (for parameters see table 3). Finally, the results were converted to polygons using the **tree_hulls** function (table 3).

	dt1	dt2	R	Zu	hmin	speed_up
trees	1.5	2	1	15	5	10000
shrubs	0.5	1	1	5	0.2	10
tree-shrub	1.5	2	2	15	0.6	10
tree-shrub (tree)	1.5	2	2	15	2	10
tree-shrub (shrub)	1.5	2	2	15	0.6	10

Table 3: *Parameters for the li2012 function*

2.4.4 uavRst

The uavRst package, introduced by Christoph Reudenbach, contains three valuable function that are of interest for the purpose of this study. Two of them were tested, since the third one uses a similar algorithms to **silva2016**

The first function is the **chmseg_FT** function. It is based on the ForestTools package and uses an approach based on the imagr watershed algorithm. From an input tree position shapefile and a CHM it calculates crown area based on a watershed algorithm. The outout format can be defined by the user. Further input variables are the minimal tree height, the radius of the moving window and a boolean input for verbosity [Meyer and Reudenbach, 2018].

Approach: The tree position layers and the CHM's were loaded. This time, the CHM's were smoothed by applying a Gaussian filter. The filter proved to yield better results in the end. Window sizes for the Gaussian filter: trees = 39; shrubs = 11; tree-shrub = 39. Further, the **chmseg_FT** function was applied to trees, shrubs and tree-shrub layer (table 4).

The second function is the **chmseg_RL** function. It is based on the rLiDAR package and is based on the generic approach of Silva et al. as described above. The advantage of the **chmseg_RL** function is its user friendliness, as

	minTreeAlt	format	winRadius	verbose
trees	2	polygons	1.5	FALSE
shrubs	0.6	polygons	1.5	FALSE
tree-shrub	0.6	polygons	1.5	FALSE
tree-shrub (tree)	2	polygons	1.5	FALSE
tree-shrub (shrub)	0.6	polygons	1.5	FALSE

Table 4: *Parameters for the chmseg_FT function*

can be stated for **chmseg**-functions. The function requires only a CHM raster and a tree position matrix. The input parameters are the maximal crown area and an exclusion factor [Meyer and Reudenbach, 2018] (see 2.4.2).

Approach: The tree position layers and the CHM's were loaded. Again, the CHM's were smoothed by applying a Gaussian filter. Window sizes for the Gaussian filter: trees = 15; shrubs = 39; tree-shrub = 15. Further, the **chmseg_RL** function was applied to trees, shrubs and a tree-shrub layer (table 5).

	maxCrownArea	exclusion
trees	10000	0.4
shrubs	5000	0.4
tree-shrub	5000	0.4
tree-shrub (tree)	5000	0.4
tree-shrub (shrub)	5000	0.4

Table 5: *Parameters for the chmseg_RL function*

2.5 CENITH

CENITH is a wrapper tool for the ForestTools package, to search the most well-suited parameters iteratively and perform the segmentation task. Its main functions are **BestSegVal** and **TreeSeg**. Furthermore, it provides a k-fold cross-validation function (TreeSegCV).

BestSegVal is a supervised approach and uses computed tree positions to validate the fitting values. It iterates of the input parameters and finds the best fitting ones. If errors are encountered, the loop continues and returns NA for the corrupted iteration. The function provides a filter parameter, that applies a sum filter to the CHM by a moving window approach. BestSegVal return a table of fitted parameters. The function requires a rasterize CHM and estimated tree positions from a vector layer.

Approach: After the CHM's and position shapefiles were loaded, the BestSegVal function was applied to all the categories (trees, shrubs, tree-shrub) (table 7).

	a	b	h	MIN	MAX	filter
trees	0.1-0.9 (by 0.1)	0.1-0.9 (by 0.1)	2-20 (by 1)	10	500000	3
shrubs	0.1-0.9 (by 0.1)	0.1-0.9 (by 0.1)	0.1-2 (by 0.1)	10	500000	3
tree-shrub	0.1-0.9 (by 0.1)	0.1-0.9 (by 0.1)	0.2-4 (by 0.2)	10	500000	3
tree-shrub (tree)	0.1-0.9 (by 0.1)	0.1-0.9 (by 0.1)	2-20 (by 0.2)	10	500000	3
tree-shrub (shrub)	0.1-0.9 (by 0.1)	0.1-0.9 (by 0.1)	0.2-5 (by 0.2)	10	500000	3

Table 6: *Parameters for the BestSegVal function*

a: numeric - single value, combination of values or sequence for MovingWindow
b: numeric - single value, combination of values or sequence for MovingWindow
h: numeric - single value, combination of values or a sequence for the maximum height of trees (in meter) to detect trees.
MIN: numeric - single value, combination of values or a sequence of minimum area for crowns. Smaller polygons are cropped Default = 0
MAX: numeric - the maximum area for crowns. Larger polygons are cropped. Default=1000

[Schönberg, 2020]

The **TreeSeg** function uses a watershed algorithm to compute polygon segments for tree crowns. The algorithm uses a moving window to identify local maxima in the rasterize CHM to compute the segments. For the computation of the tree crown segments the function requires only a rasterize CHM.

Approach: After BestSegVal yielded a table with the best-fitting values, the ones with the highest hitrate and the best over- and underestimations were inserted to TreeSeg. Using only the best hitrate proved not to be the best approach under all circumstances. Again, this function was applied to trees, shrubs, as well as to tree-shrub.

2.6 Spectral classification

For the spectral classification of the study area we exclusively used the "CAST" [Meyer, 2020] and "caret" [Kuhn, 2020] R-packages to process a RandomForest model with a four folds cross-validation for spectral image classification and a forward feature selection. The "LEGION" R-Package [Schönberg, 2020] was used to compute RGB and infrared based vegetation indices. The "RStoolbox" R-package [Leutner et al., 2019] was used to perform a Principal Component Analysis (PCA) of the generated vegetation indices.

2.6.1 Train and predict spectral classification model

Identify spectral training areas
The first step is to find suitable spectral training areas with the help of aerial RGB-image of the study area. Because we have no reliable ground truth for the study area, we are fully dependent on aerial observation. The results must be considered under this point of view as well. We selected five training areas for each class (tree, shrub, grasslands, soil, shadow).

Compute vegetation indices for training area
At first we clip the available RGB- and infrared aerial images of the study area to the extent of the generated training areas. The LEGION package was used to compute RGB (VVI, VARI, NDTI, RI, CI, BI, SI, HI, TGI, GLI, NGRDI) and infrared (NDVI, TDVI, SR, MSR) based vegetation indices for the training area. The resulting rasters are now stacked and combined with the three bands of the RGB-image.

Principal Component Analysis
The previously generated vegetation indices stack is now used to perform a Principal Component Analysis (PCA) which gathers all needed information from the relatively large raster stack and saves it to three raster-files (PCA1, PCA2, PCA3) containing all necessary information. PCA1 contains approximately 75% of the raster stack information. PCA2 and PCA3 contains approximately 10%. Additional PCA rasters were rejected because of the low percentage (<5%) of explained variables.

Merge training dataset
Now the training areas shapes need to be rasterized in order to join the data to the RGB- and previously generated PCA-rasters. The Raster stack now contains the following datasets: RGB-Red, RGB-Green, RGB-Blue, PCA1, PCA2, PCA3 and the rasterized training areas. Finally we cut the raster stack by the extent of the training areas to ensure only the selected areas are used for the RandomForest machine learning algorithm.

Train RandomForest
With the previously generated dataset we train the RandomForest machine learning algorithm with a four folds cross-validation and a forward feature selection. The produced model has a mediocre RMSE of 0.21 and is used to predict the study.

Predict study area
Following the RandomForest training we use the resulting model to predict the study area. The workflow is basically the same as seen before (see 2.6.1.2&3) except the adding of training areas. The final raster stack contains now all data of the study area (RGB-Red, RGB-Green, RGB-Blue, PCA1, PCA2, PCA3). The result is a classified raster with the classes (1 = tree, 2 = shrub, 3 = grasslands, 4 = soil, 5 = shadow). For the last step we extract the raster values from the prediction raster and assign them to the generated polygons of the five segmentation algorithms. Each polygon contains now a value (classes 1-5) of the majority of containing pixel values and can thereby be classified.

3 Results

The BestSegVal function of the CENITH package yielded the following parameters to be the best fitting ones:

	a	b	h	MIN	MAX
trees	0.1	0.5	2	10	50000
shrubs	0.2	0.8	0.5	10	50000
trees & shrubs	0.1	0.9	0.2	10	50000
tree-shrub (tree)	0.2	0.9	2	10	50000
tree-shrub (shrub)	0.6	0.7	0.2	10	50000

Table 7: *Best fitting parameters for TreeSeg according to the output of BestSegVal*

The tables 8-12 show the number of tree segments that the different algorithms yielded. Further, the number of validation points, lying inside and outside the constructed polygons are presented. The results of li2012 are not presented in the results due to its failure to yield a proper segmentation in areas with slope inclination (figure 2).

All algorithms seem to perform accurate for the segmentation of trees. All of them captured all validation points. Some algorithms, such as ForestTools and rLiDAR drew polygons matching the crown area more precisely than Dalponte and Silva (see figure 4). CENITH segmented all trees accurately and seemed to match the crown area properly.

For the segmentation of shrubs, Silva and ForestTools were able to include all validation points, while Dalponte missed 3, rLiDAR 4 and CENITH also 2. Again, all algorithms yielded comparable results (see figure 5). In this case, CENITH calculated more polygons than the other algorithms, but covered the whole (assumed) crown area very acurately compared to the other methods.

Coming to the combined layer (trees and shrubs), only Silva's algorithm segmented trees as well as shrubs correctly. While Dalponte's method missed one validation point, rLiDAR and ForestTools scored lower, with 5 and 16 missed points, respectively. CENITH performed very inaccurate for the combined layer, missing 30/47 validation points. As shown in figure 6 it segmented the trees correctly, but could not discriminate the shrubs. CENITH could be forced to hit all validation points, which would though produced more false polygons. When splitting the combined layer into trees and shrub, all algorithms perform perfect on trees. Regarding the shrubs, ForestTool did the best job, missing no validation points. Therefore, the segements are very small and do not cover the crown area accurately. Dalponte managed to include all except one (table 12). CENITH missed 21 validation points, computing noticably less polygons as the other algorithms. Figure 3 shows the estimated tree and shrub positions, as mentioned in the methods part before. Figures 4-8 show the spectral classification of the computed polygons. It has to be noticed that algorithms that computed larger crown areas (e.g. ForestTools and CENITH) are more likely to be classified as gras. The figures 9-13 show the computed segments classified by height (shrub: 0-4m; tree:>4 m) and all other classes than shrub and tree were discarded, due to unsatisfying results of the spectral classification. It can be stated that all algorithms performed very well segmenting shrubs and trees. The combined study site still invoked difficulties identifying shrub. Just Dalponte classified shrub in a reasonable manner without including trees (see figures 5,6 and 8).

Figure 2: *Polyons computed by li2012 (lidR) for trees. It failed entirely to segment any trees correctly and was therefore not taken into account in further investigation. This study site is supposed to be the easiest for the algorithms.*

algorithm	segments	inside	outside	validation points
Dalponte	122	122	3	125
Silva	125	125	0	125
rLiDAR	125	121	4	125
ForestTools	125	125	0	125
CENITH	136	123	2	125

Table 8: *Performance of the different algorithms in regards to the number of **shrub** segments created and the number of validation points lying inside and outside the computed polygons*

algorithm	segments	inside	outside	validation points
Dalponte	11	13	0	13
Silva	13	13	0	13
rLiDAR	13	13	0	13
ForestTools	12	13	0	13
CENITH	13	13	0	13

Table 9: *Performance of the different algorithms in regards to the number of **tree** segments created and the number of validation points lying inside and outside the computed polygons*

algorithm	segments	inside	outside	validation points
Dalponte	46	46	1	47
Silva	47	47	0	47
rLiDAR	47	42	5	47
ForestTools	31	31	16	47
CENITH	22	17	30	47

Table 10: *Performance of the different algorithms in regards to the number of **tree & shrub** segments created and the number of validation points lying inside and outside the computed polygons*

algorithm	segments	inside	outside	validation points
Dalponte	8	8	0	8
Silva	8	8	0	8
rLiDAR	8	8	0	8
ForestTools	8	8	0	8
CENITH	8	8	0	8

Table 11: *Performance of the different algorithms in regards to the number of **tree & shrub (tree)** segments created and the number of validation points lying inside and outside the computed polygons*

algorithm	segments	inside	outside	validation points
Dalponte	38	38	1	39
Silva	38	31	8	39
rLiDAR	39	30	9	39
ForestTools	39	39	0	39
CENITH	24	18	21	39

Table 12: *Performance of the different algorithms in regards to the number of **tree & shrub (shrub)** segments created and the number of validation points lying inside and outside the computed polygons*

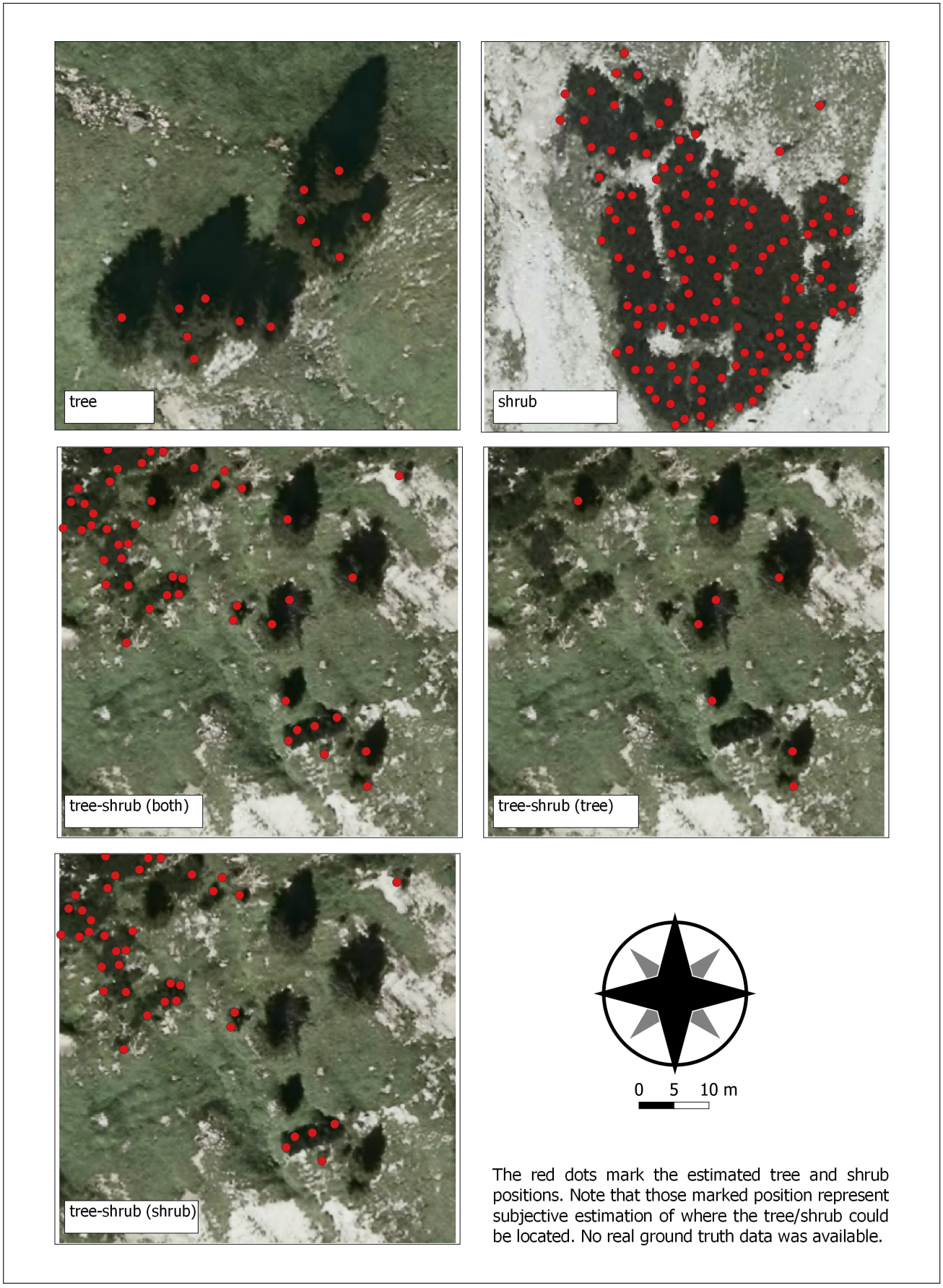

Figure 3: *Estimated tree and shrub positions for the different sites*

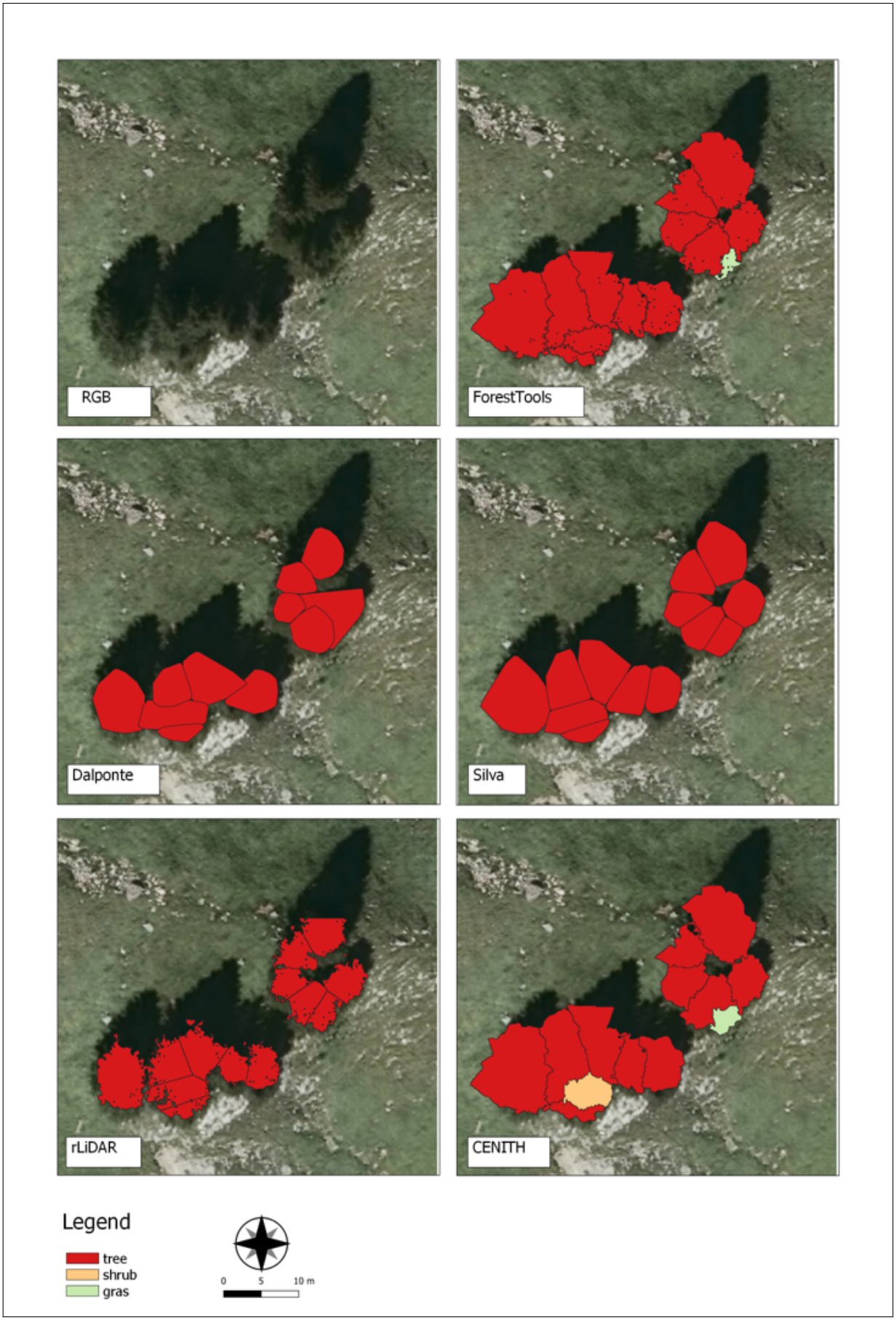

Figure 4: *Results of the segmentation algorithms on the tree study site. The polygons were classified using a RandomForest classifier to point out falsely classified segments based on a majority approach.*

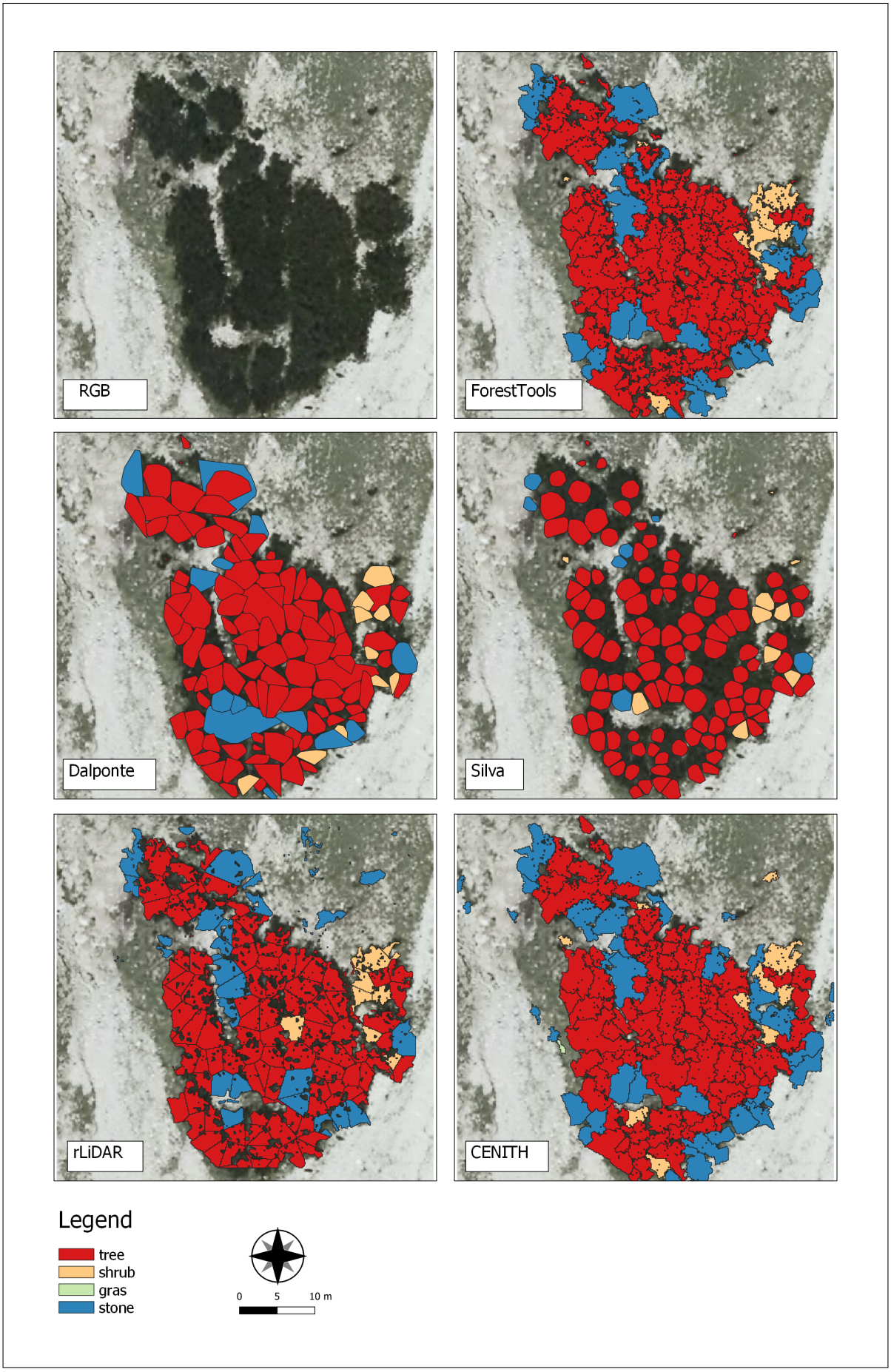

Figure 5: *Results of the segmentation algorithms on the shrub study site. The polygons were classified using a RandomForest classifier to point out falsely classified segments based on a majority approach.*

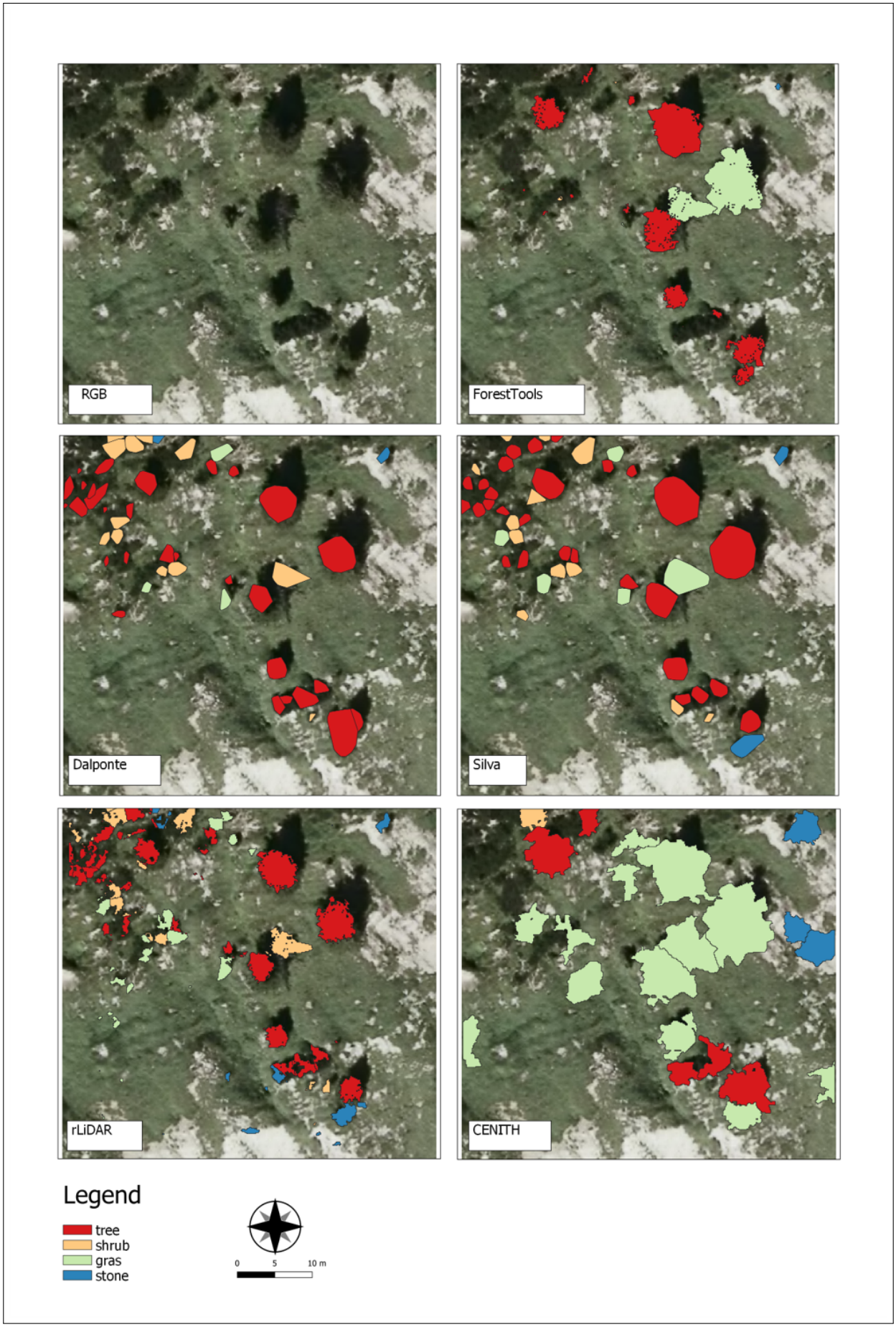

Figure 6: *Results of the segmentation algorithms on the tree-shrub study site addressing both shrubs and trees. The polygons were classified using a RandomForest classifier to point out falsely classified segments based on a majority approach.*

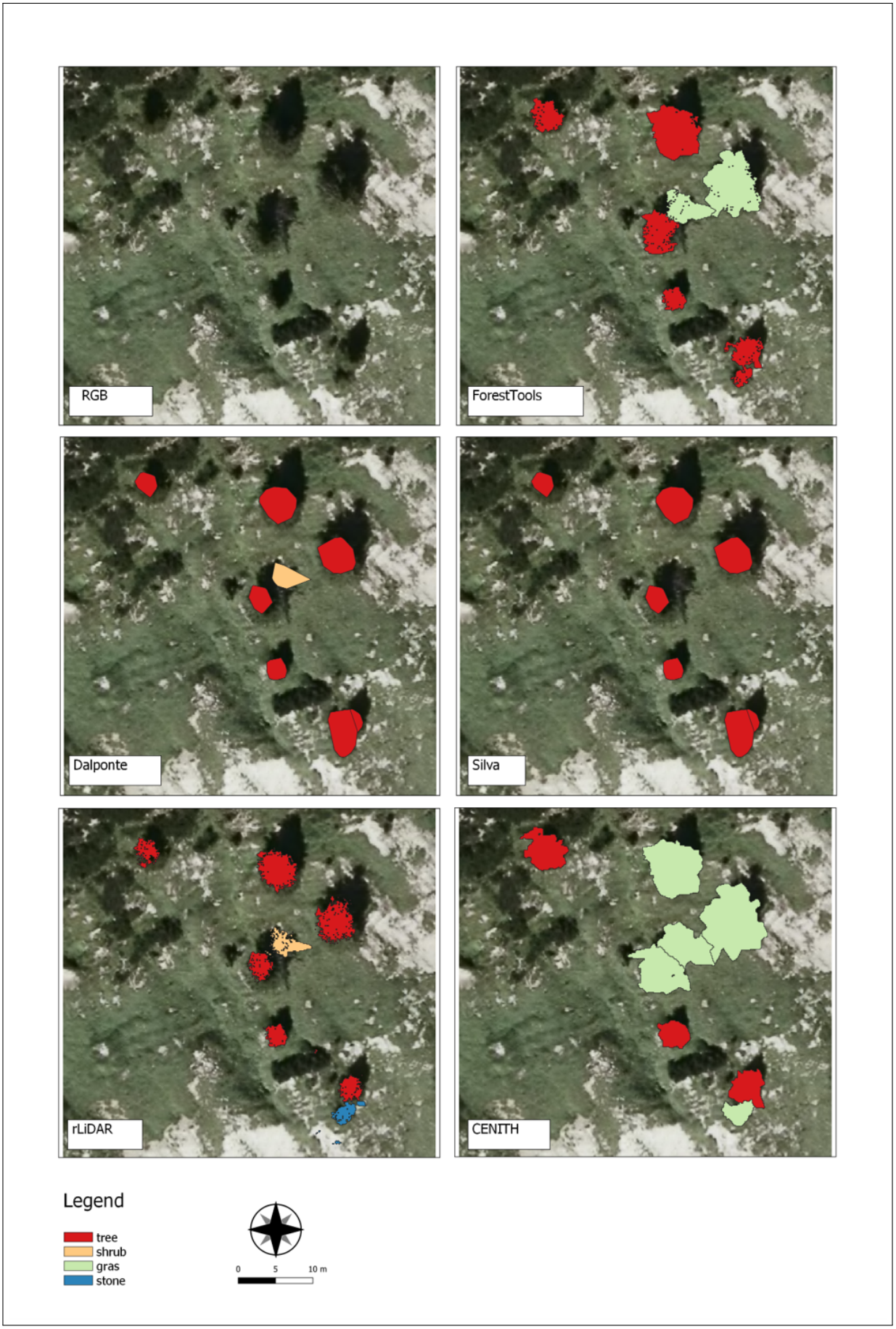

Figure 7: *Results of the segmentation algorithms on the tree-shrub study site addressing just trees. The polygons were classified using a RandomForest classifier to point out falsely classified segments based on a majority approach.*

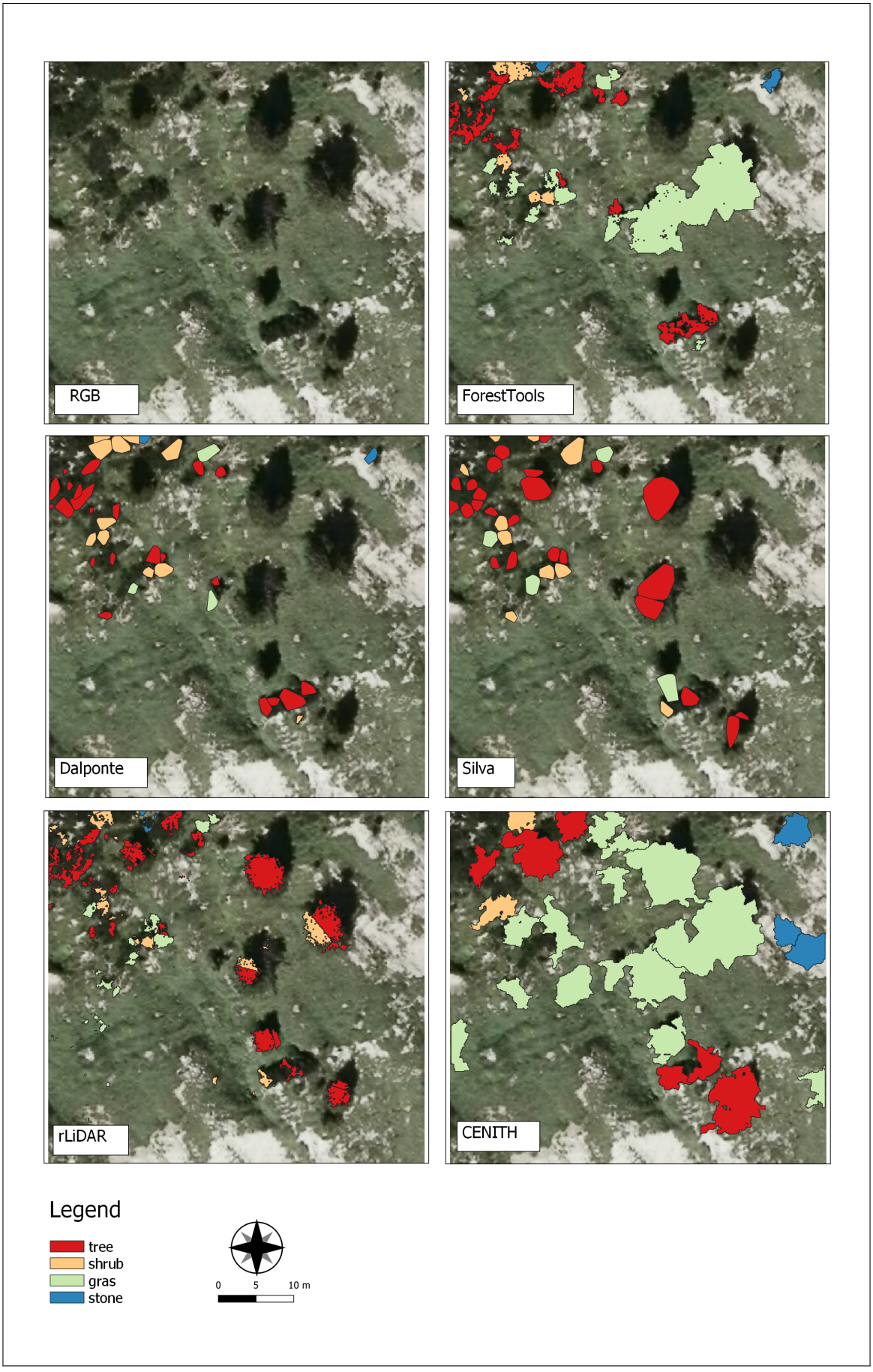

Figure 8: *Results of the segmentation algorithms on the tree-shrub study site addressing just shrubs. The polygons were classified using a RandomForest classifier to point out falsely classified segments based on a majority approach.*

Figure 9: *Results of the segmentation algorithms on the tree study site. The polygons were classified by a height of 4 m as threshold.*

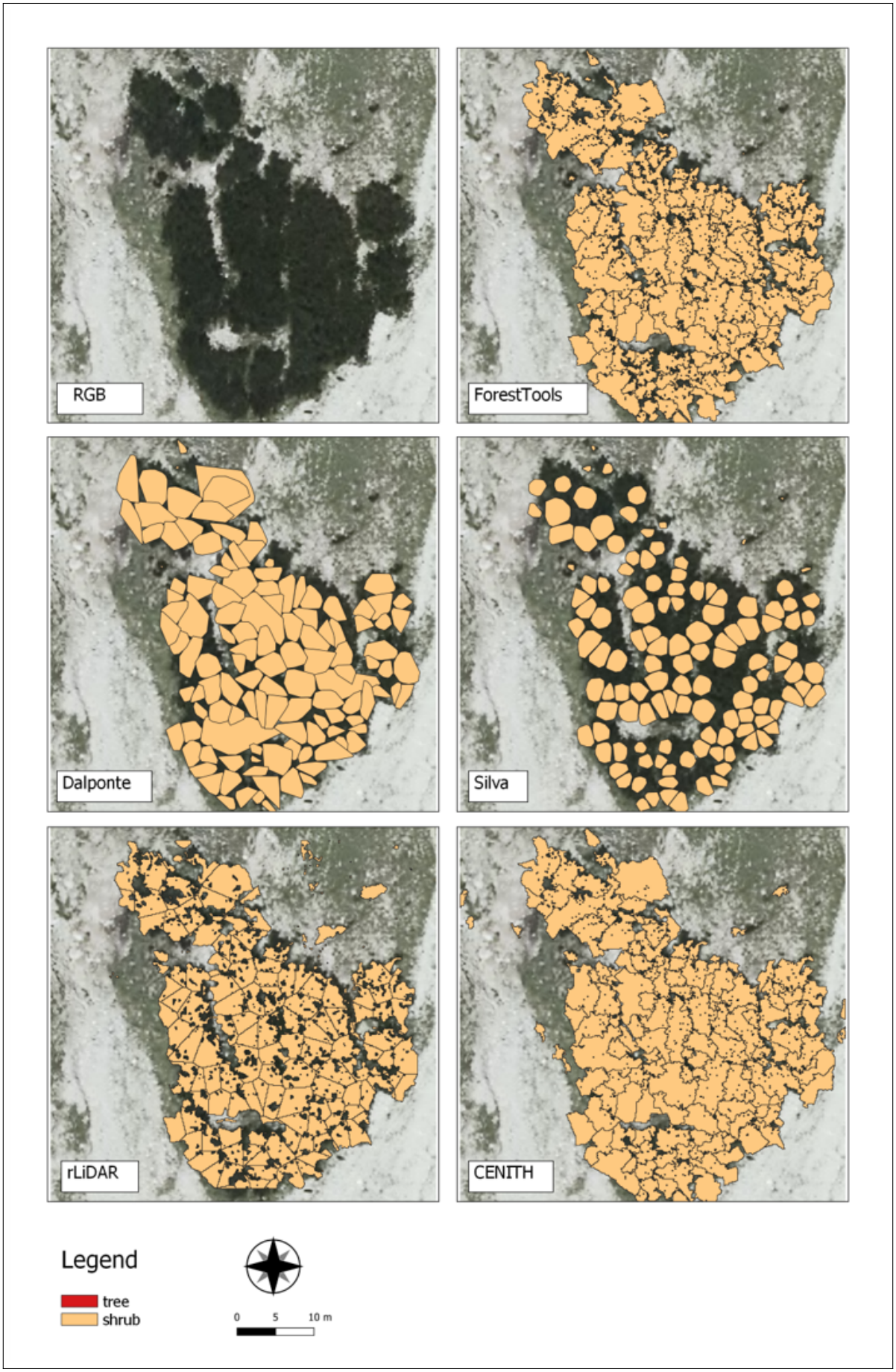

Figure 10: *Results of the segmentation algorithms on the tree study site. The polygons were classified by a height of 4 m as threshold.*

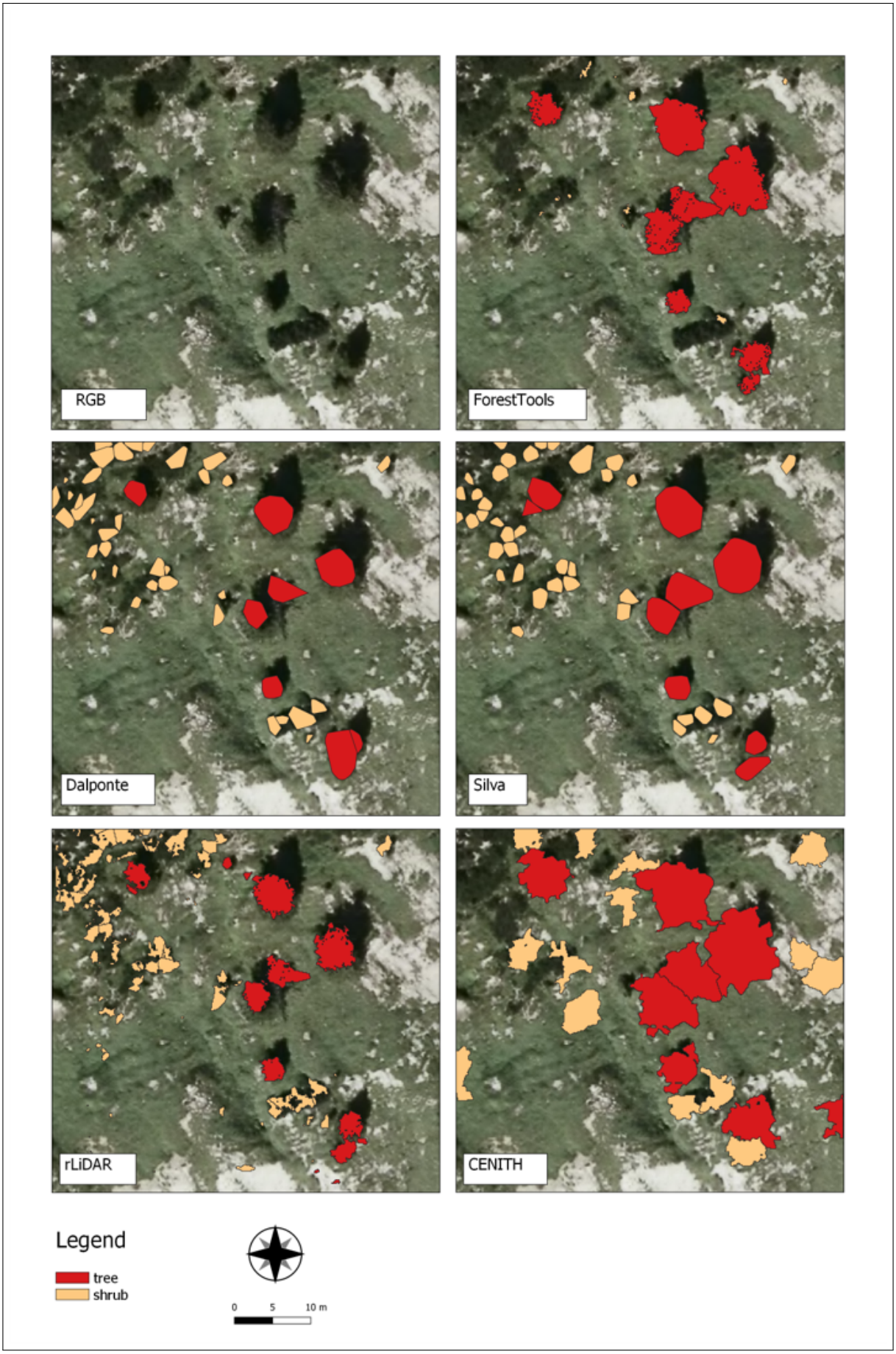

Figure 11: *Results of the segmentation algorithms on the tree study site. The polygons were classified by a height of 4 m as threshold.*

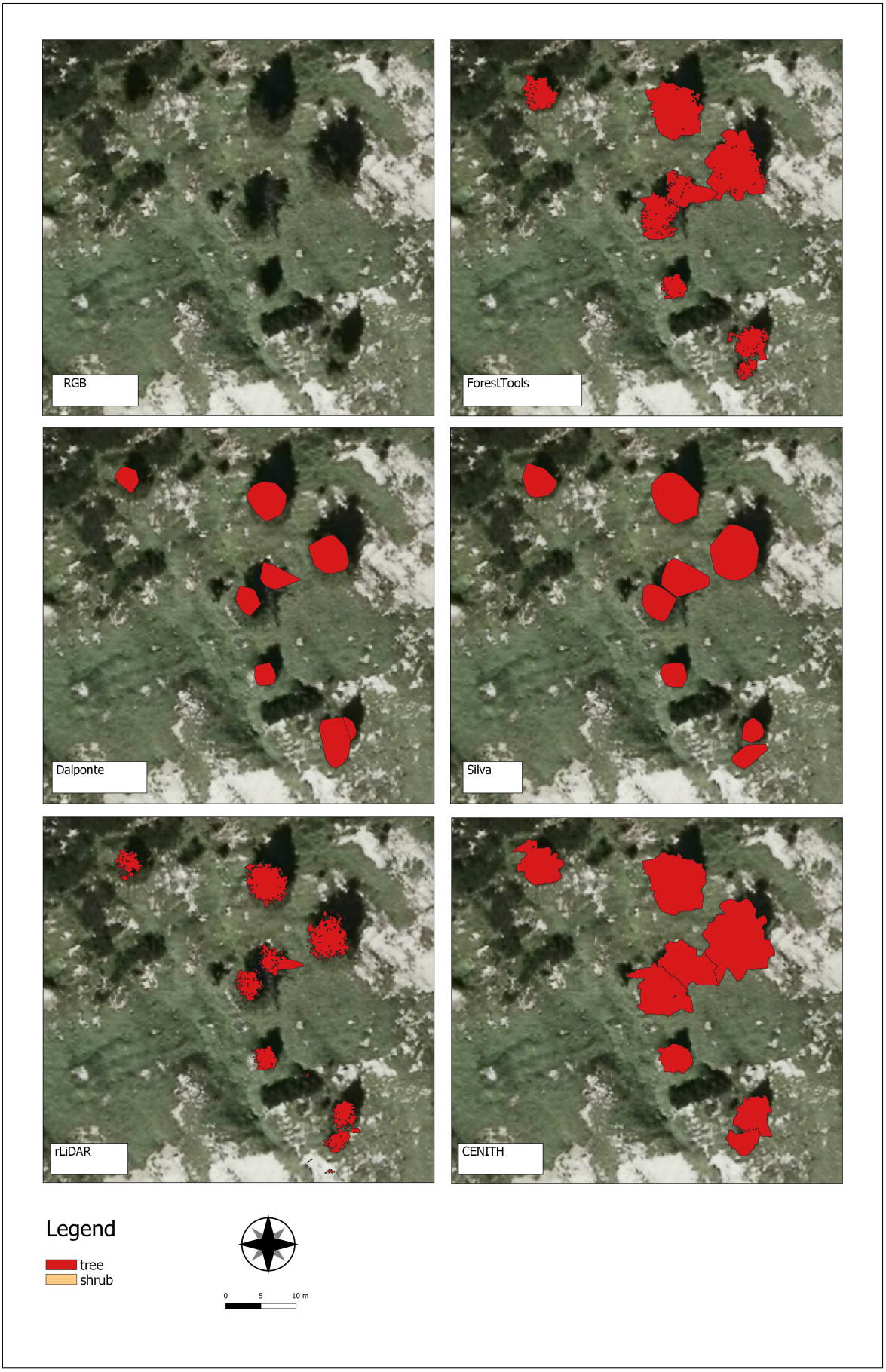

Figure 12: *Results of the segmentation algorithms on the tree study site. The polygons were classified by a height of 4 m as threshold.*

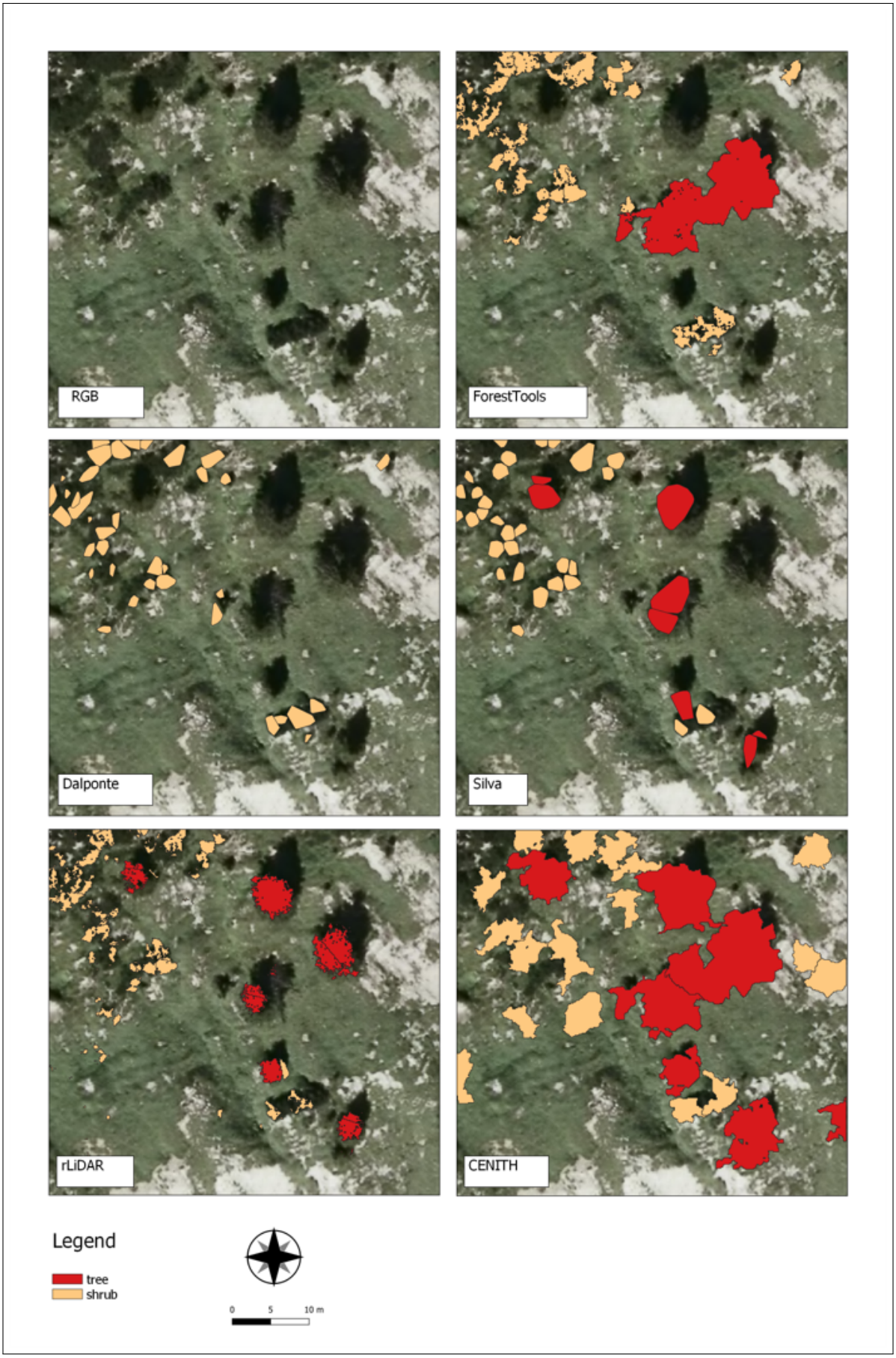

Figure 13: *Results of the segmentation algorithms on the tree study site. The polygons were classified by a height of 4 m as threshold.*

4 Discussion

CENITH performed reasonably well on the segmentation of trees and shrubs seperately. It further seems to depict the crown areas more precisely than e.g. the ITC based algorithms. The latter often just draw rough circle-like polygons. Compared to the other approaches, CENITH performed rather poorly on the shrub segmentation site. We assume that this is due to partially low data quality (i.e. shifted RGB tiles, artefacts between 5 and 200 m above ground). Since it is not supposed to be a shrub segmentation tool, rather than a tool for tree segmentation it lays within the margins of an acceptable result. Furthermore, the missing ground truth data is responsible for the poor RandomForest prediction, since shrub and krummholz of the same species could not be distinguished. Another problem is that the spectral properties of trees and shrubs within the study area are very similar and hard to differentiate using a RandomForest. It was hard to distinguish the classes tree, shrub and gras, because dark gras looks liked tree crowns in shaded areas. Additionally, it was unknown if tree and shrub are the same species but smaller or continuously damaged by avalanches, resulting in a crippled stature. Due to the questionable and rather subjectively chosen ground truth tree and shrub positions, the validity of the results of this study should be viewed highly critically. Further, the use of an LiDAR-based shannon index classification could have helped to distinguish trees and shrub patterns more accurately but the frame of the study was to narrow to include this approach additionally. Nevertheless, this study can be used as an instruction for further research.

Furthermore, the purpose of this study was to validate CENITH's applicability on the tree segmentation of alpine treelines. Since it identified all trees properly and a flawless workflow was achieved, this matter can be stated positively. Compared to the other algorithms, CENITH offers two very easy-to-use functions for the segmentation approach and one for validation. We also have to point out that we were not able to validate the cross validation fucntion of the CENITH package due to non manageable additional time expendatures. This must be testet in another approach but we assume that the results can be improved. All other tested algorithms require either more indepth knowledge for the parameterization or the use of LAS-file point cloud data. For beginners, this might be complicated and is quite RAM intensive. CENITH requires only high resolution CHM's to perform the segmentation. The other algorithms come with a clipping function, resulting in less polygons. CENITH's results could possibly be improved by implementing such a clipping function in upcoming versions.

The downside of the package is the long computation time of the BestSegVal function, which took about 1 1/2 days to find the best parameters. Nevertheless, the function can be parallelized, which decreases the computation time significantly.

5 Conclusion

All the tested algorithms yielded adequate results. Especially for the segmentation of trees all methods achieved remarkable outcomes, while diffulties were encountered in the segmentation of shrubs. The latter was mainly a problem on sites, where trees and shrubs occured equally. Under those circumstances Silva's and Dalponte's algorithms performed best, while the others, nevertheless, still yielded reasonable results. In combination with height values, even almost all supposed shrubs could be identified. The ground truth availability was the pitfall of this study, since the used validation data was rather based on subjective assumptions than on actual data. Also the built in but not tested cross validation function could prove promising in further investigation to improve CENITH's accuracy. This was unfortunately not manageable in scope of this study.

All obstacles considered, CENITH performed very well for tree segmentation. Steep slope inclination did not pose a problem, as it did for Li's algorithm [Li et al., 2012]. Since the package was not developed for shrub classification, but for tree segmentation, CENITH has met the requirements to our satisfaction. It provides an easy-to-use set of functions and its concept is easily accessible for beginners (some skills in R assumed).

References

[Allen and Walsh, 1996] Allen, T. R. and Walsh, S. J. (1996). Spatial and compositional pattern of alpine treeline, glacier national park, montana. *Photogrammetric Engineering and Remote Sensing*, 62(11):1261–1268.

[Aurenhammer and Klein, 1999] Aurenhammer, F. and Klein, R. (1999). Voronoi diagrams. in and jr sack and j. urrutia, editors, handbook of computational geometry.

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

[Guan et al., 2015] Guan, H., Yu, Y., Ji, Z., Li, J., and Zhang, Q. (2015). Deep learning-based tree classification using mobile lidar data. *Remote Sensing Letters*, 6(11):864–873.

[Harsch and Bader, 2011] Harsch, M. A. and Bader, M. Y. (2011). Treeline form–a potential key to understanding treeline dynamics. *Global Ecology and Biogeography*, 20(4):582–596.

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

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

[Kuhn, 2020] Kuhn, M. (2020). *caret: Classification and Regression Training*. R package version 6.0-86.

[Latifi et al., 2015] Latifi, H., Fassnacht, F. E., Müller, J., Tharani, A., Dech, S., and Heurich, M. (2015). Forest inventories by lidar data: A comparison of single tree segmentation and metric-based methods for inventories of a heterogeneous temperate forest. *International Journal of Applied Earth Observation and Geoinformation*, 42:162–174.

[Leutner et al., 2019] Leutner, B., Horning, N., and Schwalb-Willmann, J. (2019). *RStoolbox: Tools for Remote Sensing Data Analysis*. R package version 0.2.6.

[Li et al., 2012] Li, W., Guo, Q., Jakubowski, M. K., and Kelly, M. (2012). A new method for segmenting individual trees from the lidar point cloud. *Photogrammetric Engineering & Remote Sensing*, 78(1):75–84.

[Meyer, 2020] Meyer, H. (2020). *CAST: 'caret' Applications for Spatial-Temporal Models*. R package version 0.4.2.

[Meyer and Reudenbach, 2018] Meyer, H. and Reudenbach, C. (2018). *uavRst: Unmanned Aerial Vehicle Remote Sensing Tools*. R package version 0.5-2.

[Moore, 1979] Moore, G. K. (1979). What is a picture worth? a history of remote sensing/quelle est la valeur d'une image? un tour d'horizon de télédétection. *Hydrological Sciences Bulletin*, 24(4):477–485.

[Rahman and Gorte, 2009] Rahman, M. and Gorte, B. (2009). Tree crown delineation from high resolution airborne lidar based on densities of high points. In *Proceedings ISPRS Workshop Laserscanning 2009, September 1-2, France, IAPRS, XXXVIII (3/W8), 2009*. ISPRS.

[Reitberger et al., 2009] Reitberger, J., Schnörr, C., Krzystek, P., and Stilla, U. (2009). 3d segmentation of single trees exploiting full waveform lidar data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 64(6):561–574.

[Schönberg, 2020] Schönberg, A. (2020). *CENITH: Segmentation tool*. R package version 0.1.0.90.

[Silva et al., 2017] Silva, C. A., Crookston, N. L., Hudak, A. T., Vierling, L. A., Klauberg, C., and Cardil, A. (2017). *rLiDAR: LiDAR Data Processing and Visualization*. R package version 0.1.1.

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

[Slatyer and Noble, 1992] Slatyer, R. and Noble, I. R. (1992). Dynamics of montane treelines. In *Landscape boundaries*, pages 346–359. Springer.

[Swetnam and Falk, 2014] Swetnam, T. L. and Falk, D. A. (2014). Application of metabolic scaling theory to reduce error in local maxima tree segmentation from aerial lidar. *Forest Ecology and Management*, 323:158–167.

[Wang et al., 2012] Wang, Y., Camarero, J. J., Luo, T., and Liang, E. (2012). Spatial patterns of smith fir alpine treelines on the south-eastern tibetan plateau support that contingent local conditions drive recent treeline patterns. *Plant Ecology & Diversity*, 5(3):311–321.