‘buildenh’ - a collection of R-scripts for the cartographic enhancement of buildings generated by classification of remote sensing imagery

Joachim Höhle

Abstract

The software tool 'buildenh' enhances the classification results of remote sensing imagery by generalizing the content and vectorizing the irregular outlines of imaged objects. Vector data of urban environments are generated in an automated way and with high cartographic quality. The user may monitor the intermediate results which are displayed graphically together with the source data. The derived coordinates of the corner points may be stored together with a label in topographic databases and Geographic Information Systems (GIS). All scripts of ‘buildenh’ are open-source and the suite has the potential to contribute to national and private mapping projects and to crowd sourcing.

Keywords

Mapping, automation, vectorizing, GIS, crowd sourcing

1. Motivation and significance

Topographic map data are lacking at many places in the world [1]. Imagery from satellites, airplanes and drones is taken in large numbers and used for mapping tasks. Besides governmental organizations many private organizations support these activities using expensive equipment and highly trained human operators. There are many efforts to automate mapping and production rates. Also, amateurs should be able to carry out mapping. Such activities are summarized with the term ‘crowed sourcing’. Despite many efforts to automate topographic mapping a practical tool is still lacking [2].

The task of automated mapping of various landscape types is not trivial. Especially, mapping urban areas gives problems due to the variety of buildings and other man-made objects. The result must be vectors of high cartographic quality and geometric accuracy. The content of the map data should also be generalized. Many research projects have tried to use artificial intelligence (AI) and algorithms to automate topographic mapping. A solution of the author has been described in [3, 4]. It is based on differential rectified images using a Digital Surface Model (DSM) derived from the same imagery. The developed software package ‘buildenh’ (v1.0) achieved rectangularity and parallelism of the vectors forming the outline of buildings. The processing uses graphic displays of the intermediate results and various possibilities to edit. The geometric accuracy was tested by means of accurate reference data. From the gained experiences a new version of the software has been created. This software package, ‘buildenh’ (v1.1), is subject of this contribution.

The user of the software must prepare the processing by choosing an ortho-image and proper methods and parameters to adapt to the particularities of the imaged objects. The selections concern the type of object, the mode of processing, the partition of object, and the methods for finding the sequence of lines. The selectable parameter concerns the smallest length of a line. The user may monitor the

intermediate results which are displayed graphically together with the source data. The possibilities for editing are given by repeating the processing using another selection. Probable answers are suggested which are based on statistical methods. Furthermore, supporting scripts are given to generate a solution for very special objects. Numerous comments in the scripts help to understand the solution and make modifications easier for the user.

Automated generation of buildings and other man-made objects in vector format can be carried out by different methods. The software package 'buildenh' uses classified ortho-images, from which buildings have been extracted, as input data. The generation of vector data from the derived raster image applies the Hough transform and least-squares adjustment for the detection and processing of lines. The sequence of the detected lines in the polygon is determined by selectable methods. Successive and adjusted lines are then intersected. The coordinates of the corners are improved using least-squares adjustment. The generated outline of the building will have orthogonal and parallel lines. Lines of other orientation may be added. Tools for interactive editing are integrated in the software.

Related work of other authors regarding cartographic enhancement and regularization of classification results the work is discussed in the following. The method applied in [5] derived several mask layer data from ortho-images and DSMs and processed the building outlines by means of various image processing and GIS tools. The extracted outlines were simplified using the Douglas-Peucker algorithm [6].

In [7], the outlines of the buildings have been extracted from a derived DSM using a minimum bounding rectangle. The approach in [8] used a dense point cloud derived by image matching. Buildings are extracted by thresholding the gradients of elevations. The edge pixels of the roof are then traced, and closed polygons are generated. By a split-and-merge process, line segments are obtained, and straight lines are fitted to them. Successive lines are intersected, and roof corners are obtained.

In [9], a building mask is derived from a DSM, and then further refined by classifying geometrical features of the images. By means of a tracing algorithm, boundary points are extracted, and a set of line segments is then fitted to the extracted points by least-squares adjustment. The obtained line segments are intersected and connected to building outlines.

Other investigations start from Airborne Laser Scanning (ALS) and identify, trace, and regularize the outlines of buildings. In [10], e.g., a Delaunay triangulation is applied to an ALS point cloud. A boundary line consisting of small segments can then be identified. The lines segments are smoothed, and corners are derived. The use of two primary data sources (camera, ALS) requires simultaneous data acquisition and, therefore, more expensive equipment. The discussed work did not announce open-source software packages.

2. Software description

The R-package 'buildenh' contains nine main programs, numerous functions and supporting scripts. Their names are 'line_detection.R', 'func_mean_line.R', and 'support_line_detection.R'. Data and parameters chosen by the author are also part of the package. Details of the used data are taken from

[11, 12]. The applied parameters are adapted to these data. The diagram in Fig. 1 depicts the components of “buildenh_v1.1”. For a few objects, the processing of examples required special solutions which are contained in additional scripts, e.g., ‘spObj_line_detection.R’. Other open-source packages are “EBImage” and “spatstat”. Details about these R-packages can be found in [13, 14]. Minor R packages are ”tiff”, “rpart”, and “nlme”, which can be downloaded from [15].

2.1. Software architecture*:*

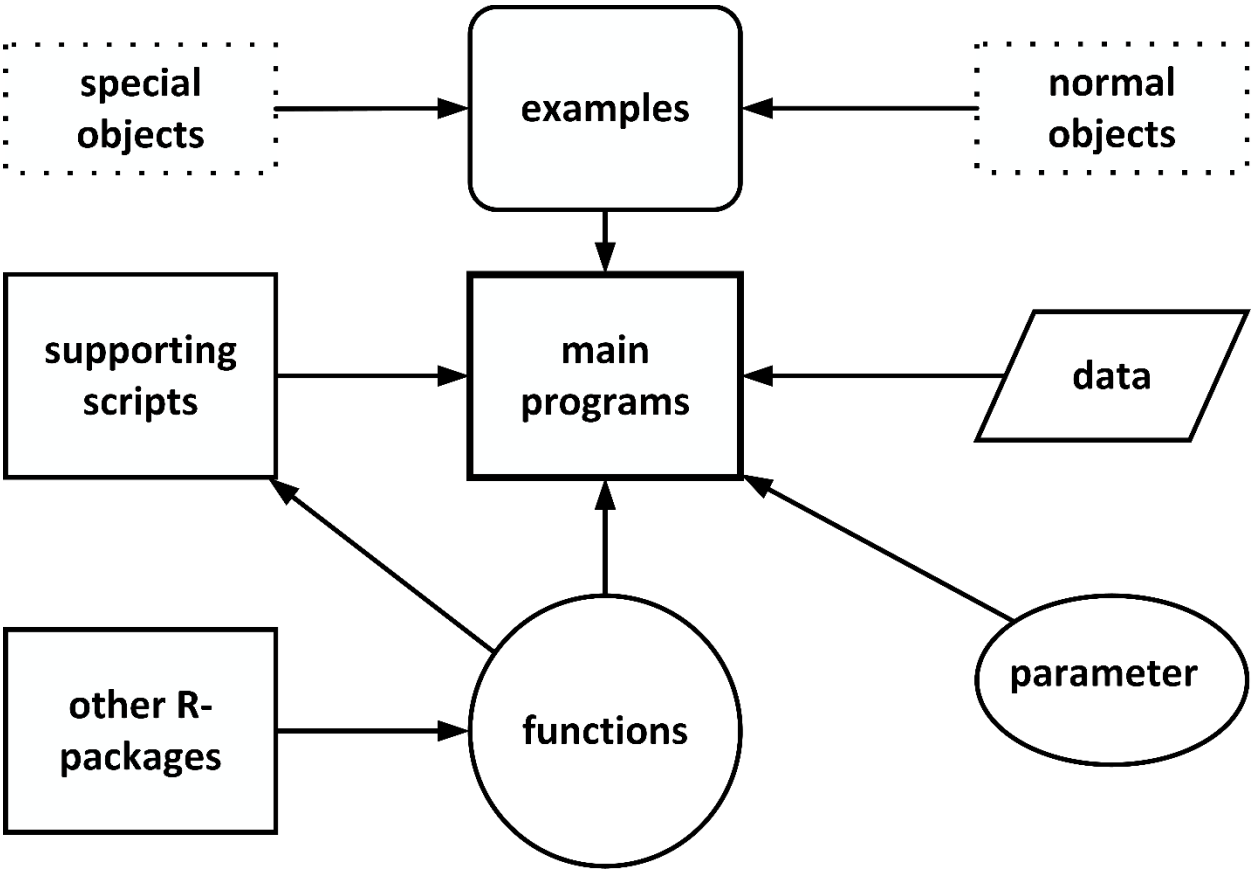

Fig. 1. Components of “buildenh”

The collection has nine main programs (*startup_buildenh.R*, *enhance_image.R*, *extract_single_building.R*, *line_detection.R*, *sequence_of_lines.R*, *adjustment_of_line.R*, *intersect_corner_points.R*, *adjustment_of_corner_coordinates.R*, *plot_results_on_references.R*). Details of these programs are given in the following.

Each program starts with a short description of the task and instructions for the use. The required data are read and intermediate results which are required by the following programs are stored. Many parts of the programs are realized as functions. Graphic output can be used. Many tests check the plausibility

of intermediate results. The main programs can be executed fully automatically one after the other. In case of problems an interaction may become necessary. Supplementing software will help in such situations. *startup_buildenh.R*

By this program the required software packages and functions are loaded. The project details (path- and file names) of two examples are read.

enhance_image.R

The ortho-image of all extracted buildings is enhanced to prepare the vectorization. Several functions of the open-source package "EBImage" are applied. Morphological functions like 'dilation' and 'erosion' are applied to make the outlines of the extracted objects smoother. The selectable parameters are the kernel size (2x2 pixels), a deviation of the average intensity value (0.01). A shape (diamond) has been used in the function 'makeBrush'. Precise edges of the outlines of the connected components (CC) representing the object (building) are generated by using the function 'thresh'. All CC are filled with pixels and labelled. Features like "area" and "maximum radius" of all objects are derived. Objects smaller than a threshold, e.g., 25 m^2 in nature, at the two used examples, are then removed in the following processing. Other scripts must be called to scale the image if required.

extract_single_building.R

Features of an ellipse which is fitted to the CC of a single building are calculated. These features (centre coordinates, azimuth of the main axis) are used in the detection of line segments and of their sequence. Plots of the CC may be overlayed onto an enlarged ortho-image. Errors in the classification may be detected. An image of the extracted single building with additional check marks for scaling is generated and stored using the R package "tiff" [17].

line_detection.R

The detection of the line segments forming the outline of the buildings uses the Hough transform [16] where the line segments are displayed as points in the parameter space (H). Default values for the resolution of H are set to $\Delta\Theta$=5° and $\Delta\rho$=5 pixels. The array of H will then be manageable by smaller computers and the line segments may be safely detected. The ρ-range should be as small as possible but contain all line segments of the object. Three possibilities for the calculation of the ρ-range are available. The default value covers the range between origin and farthest object pixel. The results of the Hough transform are ordered by the lengths of the lines. One line is selected as reference line. A plot of this line together with CC and the ortho-image will inform whether the selection may serve as reference line for other parallel and rectangular lines. Further calculations are branched for the four object types. Fig. 2 depicts the available object types in the current version of "buildenh" (v1.1). The calculation in the four branches is different. At the first two types, four lines are found by their length at extreme positions in both major directions. For the types 'multiple rectangular lines' and 'multiple rectangular and non-orthogonal lines', the solution is found by means of the functions *func_rectangular_lines* and *func_line_reduce*. The lines are analysed for their ρ- and length-values to find all line segments forming

the outline of the building. A threshold for the minimum distance between two adjacent lines (*dp*) avoids that lines are too close to each other. The selected default value is *thr* = 10 pixels.

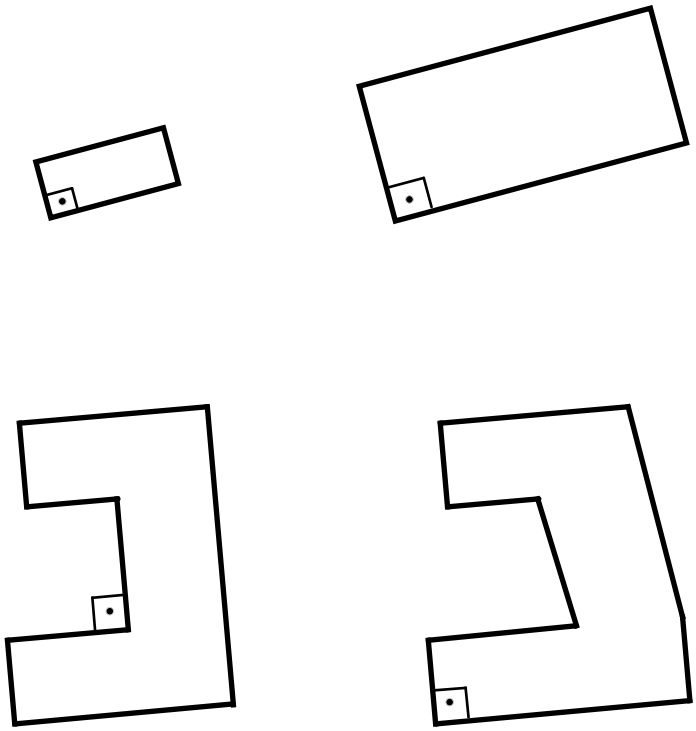

Fig. 2. Types of objects in “buildenh”. Upper row: 4-line rectangular (small and long), lower row: multiple rectangular lines and objects containing rectangular and non-orthogonal lines.

All points representing a detected line are extracted and the calculation of precise line parameters is carried out. The detected lines may contain points of another line. An analysis of the histogram of the point coordinates will detect gaps in the line segment. Points of other lines will then be removed. All detected lines and their centres are plotted on top of the ortho-image to discover whether the detected lines and their midpoints represent the imaged building correctly.

sequence_of_lines.R

The sequence of lines may be determined by means of three methods, cf. Fig. 3. The first possibility uses the angle (α) between the centre of the CC and the midpoints of the pixel cluster (PC) representing a line segment. Ordering the angles according to their size will result in the sequence. In the second solution, a calculation of the minimum distance between the midpoints of line segments may be used for more complex building shapes. A third solution searches the midpoints of the line segments. The search is carried out in a 11x11 pixels large area around the midpoints. The used functions in this program are part of the open-source package “spatstat”.

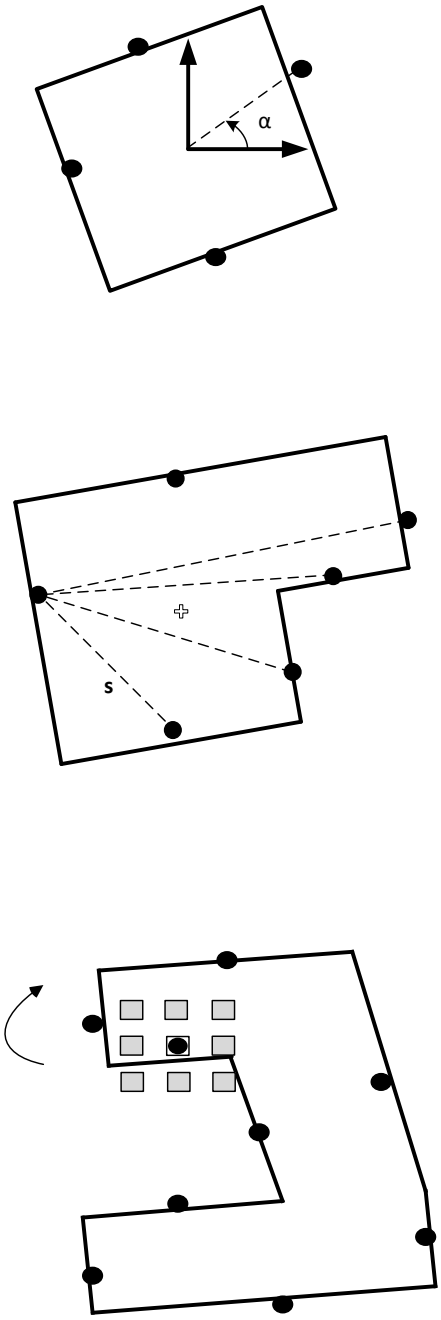

Fig. 3. Methods for finding the sequence of lines represented by their midpoints. Top: based on angle (α), middle: based on distance (s), bottom: based on scanning and line following.

A decision tree (DT) is applied for the selection of the best suited method considering the characteristics of the object. The solution is carried out by the packages “rpart” [18] and “nlme” [19] and offered to the user as suggestion.

adjustment_of_line.R

A least-squares adjustment is carried out for all detected line segments. The adjusted lines may be plotted together with the CC. The least-squares adjustment of a single line (g) minimizes the orthogonal distances (r_i), cf. Fig. 4.

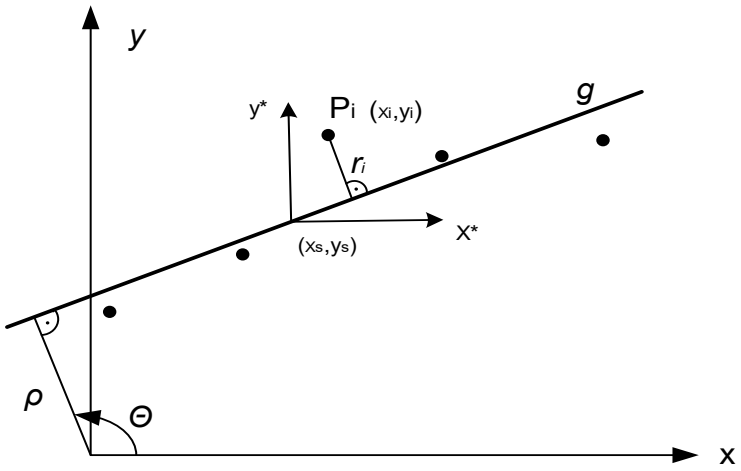

Figure 4. Orthogonal distance (r_i) from a point P_i (x_i,y_i) to a line segment (g). The black dots are pixels of the point cloud (PC) belonging to a detected line.

intersect_corner_points.R

The calculation of approximate corner points is carried out using the adjusted line parameters. The interior angles of the closed polygon are determined, and the deviations from rectangularity are compared with a threshold. If the test is successful, a weighted average for the major direction (Θ_{av}) is calculated. The approximate corner points and the connecting lines are plotted together with the CC. The object type which contains rectangular and non-orthogonal lines is treated by additional code.

adjustment_of_corner_coordinates.R

The calculation of accurate corner coordinates uses the input of the approximate coordinates and of the adjusted major orientation. The least-squares adjustment calculates new parameters (ρ_i) and the standard deviation of the residuals (r_x, r_y). A test checks if the maximum residual is less than a threshold. In this case, the final coordinates of the corner points (vertices) are calculated and stored together with their derived labels.

plot_results_on_references.R

An overview of the quality of the result can be obtained by overlaying the final vectors of a single object (building) with the DSM-based ortho-image and, if available, with a topographic map. This may be done at small or large scale. In case of an unsatisfactory result, the calculations can be repeated using another object type and/or method for the determination of the line sequence.

Supplementing scripts are provided for some of the main programs. Examples are:

-plot of numbers of buildings after applying a threshold for minimum size of area,

-interactive determination of ortho-lines and non-ortho-lines by measuring one or two pixels in an ortho-image extract, and

-plot of building outlines with corner- and line-numbers on an extract of the ortho-image.

In the given two examples, a few special objects required corrections. For example, a calculated midpoint of the line segments could be well outside the line. The position of the midpoint must be corrected using manual measurements. In the main program, the script *spObj_sequence_of_line.R* is then accessed, and the midpoint position of this line will be corrected. Another example for a special solution is the partition of objects where pixels must be measured at a position where the splitting should take place. The complete package "buildenh", version 1.1, has 116 files and 39 directories, and requires about 109 MB disk space.

2.2. Software functionalities

The classification results of DSM-based ortho-images are converted by “buildenh” into accurate and georeferenced vector data of high cartographic quality. The raster data of buildings are first enhanced with image processing methods and then processed to vectors object by object. The processing is based on extraction of straight lines using the Hough transform and least-squares adjustment of the lines and coordinates of the building corners. Only few pixels of the line must be visible in the original images. This is an advantage of our approach because vegetation often hides parts of the building. The generated outline of the buildings will be a closed polygon with parallel, rectangular, and non-orthogonal vectors. Only buildings with specifiable area and side length are processed at the given examples. The results are checked by superimposition of the generated vectors onto the DSM-based ortho-image. Errors in the automated processing may be visually recognized and interactively corrected.

3. Illustrative examples

A topographical database must fully and accurately reflect the buildings of an urban environment. An automatically or manually produced land cover map using remotely sensed imagery cannot fulfil this condition. Overlapping vegetation or other objects often hide parts of the building. This approach requires only few pixels of the imaged building edges to reconstruct a line. Furthermore, the least-squares adjustment for lines and vertex coordinates results in high geometric accuracy of the generated building representation.

Figure 5 depicts the result of the automated mapping using “buildenh” together with the DSM-based ortho-image (source data) and the automatically derive point cloud of the building outline (PC). The automated enhancement using “buildenh” generated the building polygon completely and in its true position.

Fig. 5. Enhanced outline of building (blue polygon) together with an extract from the DSM-based ortho-image and point cloud of the building outline (green dots) derived from a classification.

A result of the automated processing of several buildings is depicted in Fig. 6. The generated polygons reveal good agreement between the vectors and the imaged outline of the DSM-based ortho-image and a high cartographic quality. Additional editing may improve the thematic as well as the geometric accuracy. The quality of the result depends very much on the quality of the classification result which is not discussed in this contribution.

Fig. 6 Generated vector data together with the source data.

4. Impact

There are many attempts to generate land cover data by classification of remote sensing images. To use these data in topographic databases and Geographic Information Systems a generalization and vectorization of the raster is necessary. The software collection “buildenh” will fulfil this task. It has already been used in many tests which are published in international journals. In [3], the mathematic background of the applied methods is presented, and the geometric accuracy was evaluated in [4]. The latest version of the open-source software is available as repository at https://GitHub.com/JoaHoe/buildenh_jh. The use of the package is semi-automatic but checking and editing of intermediate results becomes always necessary at complex urban environments. Examples with other remote sensing imagery and elevation data should be carried out. Further attempts to speed-up the processing is of interest at large mapping tasks. More default values of parameters may be used to reduce the number of manual operations.

5. Conclusions

The R-software “buildenh” solves the vectorizing of imaged urban objects. Input are the classification results of DSM-base ortho-images. Output are vectors of high cartographic quality and geo-referenced coordinates of polygon vertices including a label. The package has the potential to contribute to national and private mapping projects. The tool has previously been used in two published investigations and enables further developments.

References

[1] Konecny G, Breitkopf U, Radke A. The status of topographic mapping in the world. A UNGGIM - ISPRS PROJECT 2012-2015. Int Arch Photogramm Remote Sens Spat Inf Sci 2016; XLI-B4:741-4.

[2] Li J, Huang X, Tu L, Zhang T, Wang L. A review of building detection from very high resolution optical remote sensing images. GISci & Remote Sens 2022; 59:1, 1225-26. https://doi.org/10.1080/15481603.2022.2101727 [accessed 24 October 2022]

[3] Höhle J. Generating topographic map data from classification results. Remote Sens 2017; 9: 224; https://doi.org/10.3390/rs9030224 [accessed 24 October 2022]

[4] Höhle J. Automated mapping of buildings through classification of DSM-based ortho-images and cartographic enhancement. Int J Appl Earth Obs Geoinf 2021; 95:102237. https://doi.org/10.1016/j.jag.2020.102237 [accessed 24 October 2022].

[5] Tarantino E, Figorito B. Extracting buildings from true color stereo aerial images using a decision-making strategy. Remote Sens 2011; 3:1567-14.

[6] Douglas DH, Peucker TK. Algorithms for the reduction of the number of points required to represent a digitized line or it caricature. Can Cartogr 1973; 10:122-10.

[7] Bulatov D., Rottensteiner F, Schulz K. Context-based urban terrain reconstruction from images and videos. ISPRS Ann Photogram Remote Sens Spat Inf Sci 2012; 1:190-5.

[8] Wang Y. Automatic extraction of building outline from high resolution aerial imagery. Int Arch Photogram Remote Sens Spatial Inf Sci 2016; XLI-B3 423-4.

[9] Partovi T, Bahmanyar R, Krauss T, Reinartz P. Building outline extraction using a heuristic approach based on generalization of line segments. IEEE J Sel Top Appl Earth Obs Remote Sens 2017; 10:947-14.

[10] Awrangjeb M. Using point cloud data to identify, trace, and regularize the outlines of buildings. Int J Remote Sens 2016; 37: 579-28.

[dataset] [11] ISPRS WG III/4 (2012-2016). ISPRS Test project on urban classification, 3d building reconstruction and semantic labeling, Benchmark on Semantic Labeling (isprs.org). [accessed 24 October 2022]

[dataset] [12] Gerke M. Normalized DSM. Research Gate; 2014 https://www.researchgate.net/publication/270104315_Normalized_DSM. [accessed 24 October 2022]

[13] Pau G, Fuchs F, Sklyar O, Boutros M, Huber W. EBImage-an R package for image processing with applications to cellular phenotypes. bioinform 2010; 26:7, 981-2. https://doi.org/10.1093/bioinformatics/btq046 [accessed 22 October 2022]

[14] Baddeley A, Rubak E, Turner R. Spatial point patterns: Methodology and applications with R. Chapman and Hall/CRC Press; 2015.

[15] R: The R Project for Statistical Computing (r-project.org) [accessed 22 October 2022]

[16] Hough transform Hough transform - Wikipedia [accessed 22 October 2022]

[17] tiff (R package), https://www.rforge.net/tiff/ [accessed 22 October 2022]

[18] rpart (R package), https://cran.r-project.org/package=rpart [accessed 22 October 2022]

[19] nlme (R package), https://svn.r-project.org/R-packages/trunk/nlme/ [accessed 22 October 2022]