Reproducible Extraction of Building Footprints from Airborne LiDAR Data: A Demo Paper

Mertcan Erdem
Dept. of Geomatics Engineering
Hacettepe University
Turkey
mertcanerdem@hacettepe.edu.tr

Berk Anbaroğlu
Dept. of Geomatics Engineering
Hacettepe University
Turkey
banbar@hacettepe.edu.tr

ABSTRACT

Automatic building footprint detection from aerial LiDAR point cloud data is an important research area that has use in various research areas ranging from city modeling to disaster management. Various steps must be carried out to obtain building footprints represented as polygons. Specifically, Digital Surface Model (DSM) and Digital Terrain Model (DTM) should be estimated, and then different filters (e.g. adaptive threshold, morphological open/close) and indices (e.g. terrain ruggedness index) are used to remove noise and vegetation. However, there has been lack of open-source software to realize these steps. Consequently, computational reproducibility of the findings has remained limited. The 2022 ACM SigSpatial GIS Cup contributed to this issue by openly providing aerial LiDAR point cloud data for 20 test sites, and evaluated the submissions performance by relying on a modified Intersection over Union (IoU) metric. The aim of this demo paper is to describe the development of a computationally reproducible framework to realize the GIS Cup's objective. First, the developed Python package ***lasbuildseg*** is described, and then its use on five different test sites from the GIS Cup are evaluated.

CCS CONCEPTS

• **Information Systems** → *Information Systems Applications*, *Geographic Information Systems*; **Applied Computing** → *Education*

KEYWORDS

LiDAR, Building Footprint, Computational Reproducibility

1 INTRODUCTION

Light Detection and Ranging (LiDAR) is a remote sensing technology that plays a crucial role in various research areas. Laser beams are used to measure the distances in the form of point cloud, and create precise 3D maps of the surrounding environment [13]. The potential of LiDAR has been recognized in various research areas, including surveying [18], forestry [9], landslide analysis [8], archeology [3], and autonomous driving [12]. Building footprint detection is a common task across different research areas.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike International 4.0 License.
SIGSPATIAL '23, November 13–16, 2023, Hamburg, Germany
© 2023 Copyright is held by the owner/author(s).
ACM ISBN 979-8-4007-0168-9/23/11.
https://doi.org/10.1145/3589132.3625574

Among the three main alternatives to automatically obtaining building footprints (i.e. only remotely sensed image, only point cloud, and fusion of point cloud with images), relying on aerial LiDAR surveys have recently been widely used [7]. The ability of obtaining highly accurate and detailed mapping of terrain and buildings in short amounts of time enabled the use of LiDAR in urban planning and land management [24]. However, most of the existing studies do not make the code openly available or rely on proprietary software through the process, both of which hinders computational reproducibility [6, 14].

Computational reproducibility of research findings necessitates data, software and the whole experimental setup to be openly available [5, 16]. The first step, open-data, has already been realized. Different benchmark datasets have been provided by international organizations such as ACM and ISPRS [10, 19]. Furthermore, various research groups have also embraced open-data, to maximize research impact [21]. Last, data portals play a key role, especially in terms of distribution of geographic data. Specifically, anyone can download high-resolution Digital Elevation Model (DEM) data from USGS under the 3D Elevation Program (3DEP) [20]. Although various research investigated building footprint identification from aerial LiDAR point cloud data, there has been lack of open-source software for this purpose.

The aim of this demo paper is to describe the development of the Python package available on PyPi –***lasbuildseg***– that can be used to detect building footprints from aerial point cloud data, and analyze its use on the openly available dataset of ACM GIS Cup 2022 [1]. The data, software and experimental setup of the paper is openly provided on GitHub enabling other researchers to replicate/reproduce the findings of this paper [11].

2 LASBUILDSEG DEMO

Automatic building footprint extraction of aerial LiDAR point cloud data is mainly composed of two parts: i) segmentation, and ii) outline extraction. The former step identifies the subset of the input point cloud data that belongs to buildings by first separating ground and non-ground points. Then, non-ground points are further classified into two as building and non-building (e.g. tree, pole line etc.) by employing a terrain roughness/ruggedness index [17]. The latter part then generates the building footprint from the previously determined subset of the point cloud data by constructing edges between boundary points. This step would likely produce irregular

building footprint boundary, which may then need to be regularized [2, 15]. These two parts are realized within the developed ***lasbuildseg*** Python package that can readily be downloaded. The required parameters at each step is illustrated in Figure 1.

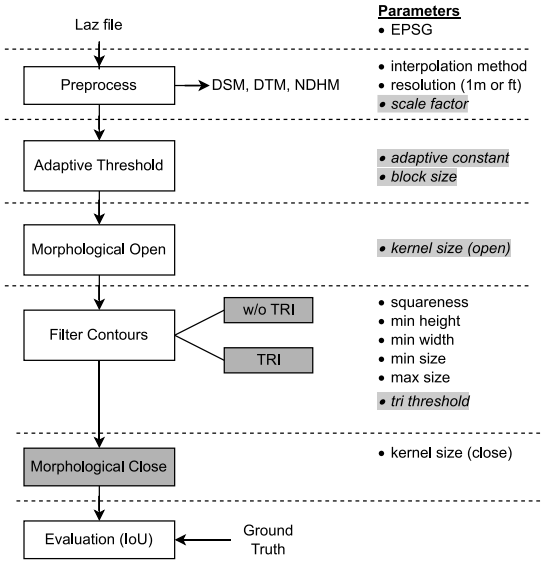


Figure 1. Reproducible extraction of building footprints from aerial LiDAR point cloud data using *lasbuildseg*. The steps in which IoU is measured, and the tuned parameters are highlighted. This framework is available on GitHub [11].

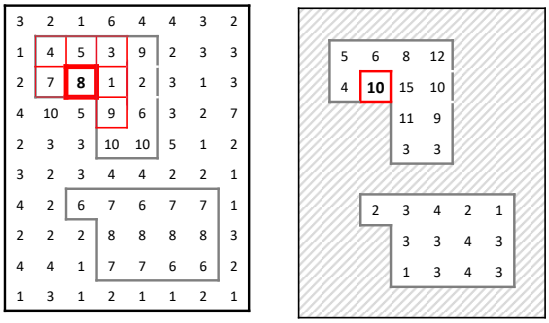
First, input is provided as a laz file. Although the Python package laspy 2.0.0 could retrieve parts of its metadata, the EPSG code could not be obtained. Therefore, the metadata of the laz file should first be identified (e.g. by using QGIS), and then its EPSG code should be provided. This input is necessary when projecting the detected building footprints into the EPSG code of the ground truth data (i.e. 3857 for the ACM GIS Cup).

The Preprocess step involves the estimation of the of the Digital Surface Model (DSM), Digital Terrain Model (DTM). The three parameters involved in this process are: i) interpolation method, ii) resolution, and the iii) scale factor. The three different interpolation methods available are, i) *nearest*, ii) *cubic*, and iii) *linear*. The default value of this parameter is *nearest*. The resolution parameter defines the granularity of the sampling intervals while estimating the models. The default value is one, and whether the resolution being 1m or 1ft depends on the EPSG code. While generating the DTM, it is possible to increase the contrast of pixels by a scale factor [22]. The Normalized Digital Height Model (NDHM = DSM – DTM) is used when computing the adaptive threshold.

The adaptive threshold step removes the noise and generates the contours (i.e. potential building footprints) by relying on two parameters: i) *adaptive constant*, and ii) *block size*. This step generates a binary image, where those pixels higher than their neighborhood (specified with the block size parameter) are removed. However, this step fails when there is dense vegetation in localized parts of the study area.

The *Morphological Open* step is used to remove small features or to smooth and simplify the contours. The required parameter *kernel size* determines the neighborhood of a pixel to be considered while smoothing. Feature removal is carried out, in which a larger kernel size will result in more extensive removal of small features, while a smaller one will retain more details. Consequently, this step further refines the detected contours.

The *Filter Contours* step is important as it is the final step in terms of refining the potential building footprints. Specifically, those contours that have an area smaller than the specified min size or greater than the specified max size are removed. Similarly, the squareness threshold identifies how close the detected contour is to a square by exploring the ratio between the height and width of the contour, both of which should be greater than a threshold. The terrain ruggedness index (tri) is used to identify the heterogeneity within a contour. Setting the tri threshold optimally may therefore improve the accuracy of the outcome [17]. In order to understand its effect, the analysis was classified into two as before employing tri, and after employing tri. The calculation of tri is described in Figure 3. For example, the tri of the pixel outlined in red is computed as $tri = \sqrt{\sum_i (8-i)^2}$, where pixel i is one of the 3×3 neighbors (outlined in light-red) of the pixel. Once all the tri values are computed, their average is obtained to represent the tri of the contour. The contour at the bottom half of the image is more likely to correspond to a building footprint as the heterogeneity of its composing pixels is lower.



(a) (b)

Figure 2. Exemplar DSM with two contours outlined in gray (a), and the terrain ruggedness index values (b)

The *Morphological Close* step is used to obtain more smooth contours that are more likely to represent real building footprints. This step requires only on the kernel size (close) parameter, which is used to generate the neighborhood pixels accordingly. The default value of this parameter is three.

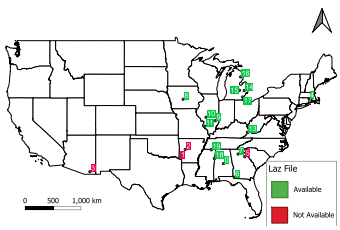
The *Evaluation* step relies on the extended IoU metric that penalizes false positives [23]. Given this metric, the ACM GIS Cup 2022 competition determined the first [19], second [23], and the third submissions [4].

An interactive demo session is successfully carried out on a group of students on different test sites illustrated in Figure 3a. In this

way, students explored ways in which to improve the performance and investigate the effect of the chosen parameters.

3 CASE STUDY

This section first describes the test sites, and the results obtained by analyzing them. The map of the test sites is illustrated in Figure 3a. The point cloud data in laz format was not available for some of the test sites as of 21 May 2023 from the competition's web-site [1]. Test sites differ in terms of point density d (pts/m²), total number of buildings in the ground truth data (|B|), and the minimum and maximum area of a building footprint (m²). The three sites are sites (i.e. sites 9, 13, and 15) have unique properties such as having the minimum area, max density and number of buildings, and max area respectively. The other two sites (i.e. 8 and 19) are chosen to cover the different regions of the available test sites.



(a)

id	*d*	|B|	min	max
8	4	40	13	476
9	7	89	*3*	1290
13	*12*	*105*	11	742
15	*3*	42	9	*1517*
19	5	70	10	617

(b)

Figure 3. Available test sites (a) and the investigated ones (b)

Various parameters could be tuned within the developed Python package ***lasbuildseg***. Nevertheless, five of these parameters have more distinguishing effects on the overall results, which are: i) scale factor $s \in \{9, 300, 1200\}$ [22], ii) adaptive constant $a \in \{2.6, 3.6,.., 6.6\}$, iii) block size $b \in \{41, 51,.., 91\}$, iv) kernel size (open) $k \in \{3, 4, 5, 6\}$, and the v) tri threshold $\in \{1, 2, 3, 4\}$.

An empirical analysis had been carried out to determine the parameter values as illustrated in the Table 1.

Table 1. Parameters of the test sites

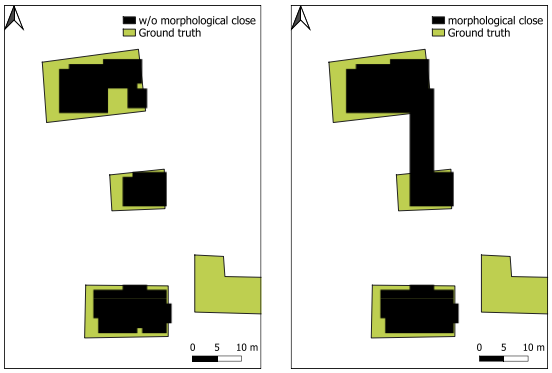
Test Site	*a*	*b*	*k*	*tri*
8	3.6	91	4	1
9	3.6	51	3	3
13	2.6	41	3	1
15	3.6	51	3	3
19	6.6	61	6	1

The effect of scale factor on the modified IoU metric for the investigated three steps (i.e. without terrain ruggedness index, with terrain ruggedness index, and the morphological close) are illustrated in Table 2. These three steps are denoted with S1, S2 and S3 respectively, and correspond to gray-filled rectangles in Figure 1. After each step, the resulting contours were converted into building footprints represented as a GeoJSON file. The provided script by the GIS Cup was used to assess these outcomes against the ground-truth data.

Table 2. The IoU scores of the experimental setup

	Test sites				
	8	**9**	**13**	**15**	**19**
$s = 9$					
S1	0.02	0.03	0.02	0.08	0.13
S2	0.12	0.11	0.25	0.21	0.14
S3	0.13	0.18	0.28	0.26	0.15
$s = 300$					
S1	0.04	0.37	0.3	0.18	0.24
S2	0.21	0.32	0.53	0.49	0.26
S3	0.25	0.35	0.55	0.54	0.25
$s = 1200$					
S1	0.09	0.4	0.21	0.27	0.25
S2	0.35	0.31	0.53	0.49	0.26
S3	0.34	0.34	0.55	0.54	0.26

The results reveal notable insights into a better understanding on the effect of parameters on different test sites. For example, morphological close operation (S3) usually increases the performance of the solution by completing the missing parts of a building footprint (as illustrated in the bottommost building footprint in Figure 4) and forms a more valid building footprint. However, in some cases it may reduce the IoU as well. Specifically, the top two building footprints in the same figure are close enough, which led to an erroneous merge after the morphological close operation.



(a) (b)

Figure 4. Limitation of morphological close (test site 8 with *s*=1200). Before closure (a), after closure (b)

A similar intriguing outcome is that although employing terrain ruggedness index (S2) usually increases the performance, it may rarely decrease it as well. For example, in the situation illustrated in Figure 5, a house is next to a tall tree which is captured within the contour. This causes height changes within the contour, which may be eliminated after employing the terrain ruggedness index. Specifically, the contour detected after applying the adaptive threshold and morphological opening (i.e. step S1) was removed when terrain ruggedness index is used. Similar situations also occur when a building has multiple-level roofs. This outcome suggests the importance of fine-tuning the tri threshold. Nevertheless, such cases are rare, and employing the terrain ruggedness index (S1 → S2) usually increases the performance substantially.

(a) (b) (c)

Figure 5. Limitation of terrain ruggedness index (test site 9 with *s*=300). A tall tree is next to the house (a). The corresponding DSM (b). The building footprint is detected *only when* terrain ruggedness index is *not* used (c).

4 CONCLUSION

This demo paper describes a computationally reproducible framework to detect building footprints from aerial LiDAR point cloud data. It relies on the developed Python package ***lasbuildseg***, and operates on the data provided by ACM GIS Cup 2022. The package relies on a filtering-based approach and improves the IoU performance iteratively with the following steps. First, adaptive threshold and morphological opening is used to remove vegetation and noise. Second, terrain ruggedness index is used to further refine the contours that have substantial heterogeneity. Finally, a morphological close operation is carried out to form more regular building footprints. The findings suggest that the aforementioned steps usually improve the IoU metric. However, fine-tuning of the parameters are required. The contents of this paper are computationally reproducible (tested on three graduate students), which may be used to enhance the quality of education and research on aerial LiDAR point cloud data analysis.

The future work would focus on the improvement of ***lasbuildseg*** in several different aspects. First, it should be tested on buildings having holes. Second, different metrics including completeness, correctness and quality should also be incorporated into the package to have a better understanding on the performance of the building footprint detection methods. Third, detected building footprints should be regularized. At the moment, each building footprint contains a large number of vertices representing its corners due to the raster-to-vector conversion. Finally, considering the *return number* of points might improve the performance, especially in situations where buildings and trees are very close as illustrated in Figure 5.

ACKNOWLEDGMENTS

The authors would like to express their gratitude to the organizers of the ACM SigSpatial GIS Cup 2022, particularly Yaron Kanza, as well as Ümmü Sude Yıldırım for her initial contributions to this research.

REFERENCES

[1] ACM SIGSPATIAL GIS Cup 2022: 2022. *https://sigspatial2022.sigspatial.org/giscup/*. Accessed: 2023-05-16.

[2] Awrangjeb, M. 2016. Using point cloud data to identify, trace, and regularize the outlines of buildings. *International Journal of Remote Sensing*. 37, 3 (Feb. 2016), 551–579. DOI:https://doi.org/10.1080/01431161.2015.1131868.

[3] Canuto, M.A. et al. 2018. Ancient lowland Maya complexity as revealed by airborne laser scanning of northern Guatemala. *Science*. 361, 6409 (Sep. 2018), eaau0137. DOI:https://doi.org/10.1126/science.aau0137.

[4] Chen, W. et al. 2022. Deep semantic segmentation for building detection using knowledge-informed features from LiDAR point clouds. *Proceedings of the 30th International Conference on Advances in Geographic Information Systems* (New York, NY, USA, 2022), 1–4.

[5] Freire, J. et al. 2012. Computational reproducibility: state-of-the-art, challenges, and database research opportunities. *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 2012), 593–596.

[6] Gilani, S.A.N. et al. 2016. An Automatic Building Extraction and Regularisation Technique Using LiDAR Point Cloud Data and Orthoimage. *Remote Sensing*. 8, 3 (Mar. 2016), 258. DOI:https://doi.org/10.3390/rs8030258.

[7] Huang, J. et al. 2019. Automatic building extraction from high-resolution aerial images and LiDAR data using gated residual refinement network. *ISPRS Journal of Photogrammetry and Remote Sensing*. 151, (May 2019), 91–105. DOI:https://doi.org/10.1016/j.isprsjprs.2019.02.019.

[8] Jaboyedoff, M. et al. 2012. Use of LIDAR in landslide investigations: a review. *Natural Hazards*. 61, 1 (Mar. 2012), 5–28. DOI:https://doi.org/10.1007/s11069-010-9634-2.

[9] Khosravipour, A. et al. 2016. Generating spike-free digital surface models using LiDAR raw point clouds: A new approach for forestry applications. *International Journal of Applied Earth Observation and Geoinformation*. 52, (Oct. 2016), 104–114. DOI:https://doi.org/10.1016/j.jag.2016.06.005.

[10] Kölle, M. et al. 2021. The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and Multi-View-Stereo. *ISPRS Open Journal of Photogrammetry and Remote Sensing*. 1, (Oct. 2021), 100001. DOI:https://doi.org/10.1016/j.ophoto.2021.100001.

[11] LasBuildSeg Building Footprint Extraction from LiDAR: 2023. *https://github.com/MertcanErdem/LasBuildSeg*. Accessed: 2023-05-30.

[12] Maddern, W. et al. 2017. 1 year, 1000 km: The Oxford RobotCar dataset. *The International Journal of Robotics Research*. 36, 1 (Jan. 2017), 3–15. DOI:https://doi.org/10.1177/0278364916679498.

[13] Mallet, C. and Bretar, F. 2009. Full-waveform topographic lidar: State-of-the-art. *ISPRS Journal of Photogrammetry and Remote Sensing*. 64, 1 (Jan. 2009), 1–16. DOI:https://doi.org/10.1016/j.isprsjprs.2008.09.007.

[14] Muftah, H. et al. 2022. Towards open-source LOD2 modelling using convolutional neural networks. *Modeling Earth Systems and Environment*. 8, 2 (Jun. 2022), 1693–1709. DOI:https://doi.org/10.1007/s40808-021-01159-8.

[15] Ozdemir, E. et al. 2022. A novel algorithm for regularization of building footprints using raw LiDAR point clouds. *Geocarto International*. 37, 25 (Dec. 2022), 7358–7380. DOI:https://doi.org/10.1080/10106049.2021.1974104.

[16] Piccolo, S.R. and Frampton, M.B. 2016. Tools and techniques for computational reproducibility. *GigaScience*. 5, 1 (Dec. 2016), s13742-016-0135–4. DOI:https://doi.org/10.1186/s13742-016-0135-4.

[17] Riley, S. et al. 1999. A Terrain Ruggedness Index that Quantifies Topographic Heterogeneity. *Internation Journal of Science*. 5, (Jan. 1999), 23–27.

[18] Sohn, G. and Dowman, I. 2007. Data fusion of high-resolution satellite imagery and LiDAR data for automatic building extraction. *ISPRS Journal of Photogrammetry and Remote Sensing*. 62, 1 (May 2007), 43–63. DOI:https://doi.org/10.1016/j.isprsjprs.2007.01.001.

[19] Song, H. and Jung, J. 2022. Challenges in building extraction from airborne LiDAR data: ground-truth, building boundaries, and evaluation metrics. *Proceedings of the 30th International Conference on Advances in Geographic Information Systems* (New York, NY, USA, 2022), 1–4.

[20] Stoker, J.M. et al. 2016. Evaluation of Single Photon and Geiger Mode Lidar for the 3D Elevation Program. *Remote Sensing*. 8, 9 (Sep. 2016), 767. DOI:https://doi.org/10.3390/rs8090767.

[21] Sun, P. et al. 2020. Scalability in Perception for Autonomous Driving: Waymo Open Dataset. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Jun. 2020), 2443–2451.

[22] Vu, T.T. et al. 2009. Multi-scale solution for building extraction from LiDAR and image data. *International Journal of Applied Earth Observation and Geoinformation*. 11, 4 (Aug. 2009), 281–289. DOI:https://doi.org/10.1016/j.jag.2009.03.005.

[23] Xu, X. 2022. An unsupervised building footprints delineation approach for large-scale LiDAR point clouds. *Proceedings of the 30th International Conference on Advances in Geographic Information Systems* (New York, NY, USA, 2022), 1–4.

[24] Yan, W.Y. et al. 2015. Urban land cover classification using airborne LiDAR data: A review. *Remote Sensing of Environment*. 158, (Mar. 2015), 295–310. DOI:https://doi.org/10.1016/j.rse.2014.11.001.